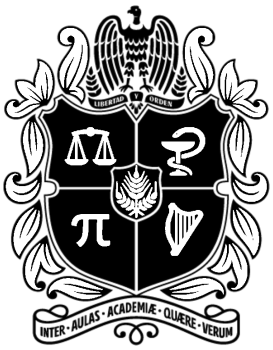
Agglomerative Hierarchical Clustering Algorithm for Community Detection in Large-Scale Complex Networks

Eduar Moisés Castrillo Velilla
System Engineering and Computing, Master (c)



Universidad Nacional de Colombia
Facultad de Ingeniería
Departamento de Ingeniería de Sistemas y Computación
Bogotá, D.C.
October 2018

Agglomerative Hierarchical Clustering Algorithm for Community Detection in Large-Scale Complex Networks

Eduar Moisés Castrillo Velilla
System Engineering and Computing, Master (c)

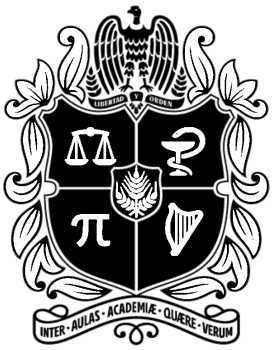
Thesis presented in partial fulfillment of the
requirements for the degree of
Master in Computer and Systems Engineering

Advisor
Elizabeth León Guzmán
Ph.D. in Computer Science

Coadvisor
Jonatan Goméz Perdomo
Ph.D. in Computer Science

Research line
Graph Mining

Research group
MIDAS



Universidad Nacional de Colombia
Facultad de Ingeniería
Departamento de Ingeniería de Sistemas y Computación
Bogotá, D.C.
October 2018

Title in English

Agglomerative Hierarchical Clustering Algorithm for Community Detection in Large-Scale Complex Networks

Título en español

Algoritmo de Agrupamiento Jerárquico Aglomerante para Detección de Comunidades en Redes Complejas de Gran Escala

Abstract: In this thesis several algorithms are proposed to compute efficiently high quality community structure in large-scale complex networks. First, a novel similarity measure that determines the structural similarity in a graph by dynamically diffusing and capturing information beyond the immediate neighborhood of connected nodes. This new similarity is modeled as an iterated function that can be solved by fixed point iteration in super-linear time and memory complexity, so it is able to analyze large-scale graphs. In order to show the advantages of the proposed similarity in the community detection task, we replace the local structural similarity used in the SCAN algorithm with the proposed similarity measure, improving the quality of the detected community structure and also reducing the sensitivity to the parameter ϵ. Second, a novel fast heuristic algorithm for multi-scale and hierarchical community detection inspired on an agglomerative hierarchical clustering technique. This algorithm uses the Dynamic Structural Similarity in a heuristic agglomerative hierarchical algorithm, that does not merge only clusters with maximal similarity as in the classical hierarchical approach, but merges any cluster that does not meet a community definition passed by parameter with its most similar adjacent clusters. The algorithm computes the similarity between clusters at the same time is checking if each cluster meets the specified community definition. It is done in linear time complexity in terms of the number of cluster in the iteration. Since a complex network is a sparse graph, this approach has a super linear time complexity with respect to the size of the input in the average case scenario, making it suitable to be applied on large-scale complex networks. Third, an efficient algorithm to detect fuzzy and crisp overlapping community structure. This algorithm leverages the disjoint community structure generated by the heuristic algorithm proposed above. Three core elements have been proposed to compute the overlapping community structure: *i)* A connectivity function that quantifies the density of connections of a node towards a disjoint community, that relies its computation on the Dynamic Structural Similarity measure. *ii)* An ϵ-Core community definition that increases the probability of identifying in-between communities in the disjoint community structure. *iii)* A membership function to compute the soft partition from the core disjoint communities. Because this algorithm keeps the same computational complexity of the original disjoint algorithm, it is still applicable to large-scale graphs. Finally, an extensive experimentation is performed in order to test the properties, efficiency and efficacy of the proposed algorithms and to compare them with

the state-of-the-art. The experimental results show that the proposed algorithms provide better trade-off among the quality of the detected community structure, computational complexity and usability, compared to the state-of-the-art.

Resumen: En esta tesis se proponen varios algoritmos para computar eficientemente estructura de comunidad de alta calidad en redes complejas de gran escala. Primero, se propone una nueva medida que determina la similitud estructural en un grafo mediante la difusión y captura de información mas allá de la vecindad inmediata de los nodos conectados que están siendo analizados. Esta nueva similitud está modelada como una función iterada que puede ser calculada por iteración a punto fijo en complejidad de tiempo y memoria super-lineal, por lo tanto puede utilizarse para analizar grafos de gran escala. Para mostrar las ventajas de la similitud estructural propuesta, se ha reemplazado la similitud estructural local utilizada en el algoritmo SCAN, con la similitud estructural dinámica, mejorando así la calidad de la estructura de comunidad detectada y también reduciendo la sensibilidad al parámetro ϵ. Segundo, se propone un algoritmo heurístico novedoso para detección de comunidades jerárquicas multi-escala que está inspirado en una técnica de agrupamiento jerárquica aglomerante. Este algoritmo utiliza la similitud estructural dinámica en un algoritmo heurístico jerárquico aglomerante, que no une solamente las comunidades con máxima similitud tal como en la técnica jerárquica clásica, sino que une cualquier comunidad que no cumple una definición de comunidad pasada como parámetro, con sus comunidades vecinas con las cuales presenta mayor similitud. El algoritmo computa la similitud entre las comunidades a la vez que verifica si cumplen la definición de comunidad pasada como parámetro. Esto es hecho en tiempo lineal en términos del número de comunidades en la iteración. Ya que una red compleja es un grafo disperso, esta aproximación presenta una complejidad de tiempo super-lineal en el caso promedio con respecto al tamaño del grafo entrada, por lo tanto puede ser aplicada en redes complejas de gran escala. Tercero, se propone un algoritmo novedoso para detectar estructura de comunidad superpuesta, tanto difusa como nítida. Este algoritmo utiliza la estructura de comunidad disyunta generada por el algoritmo heurístico propuesto anteriormente. Se proponen tres componentes principales para computar la estructura de comunidad superpuesta. *i)* Una función de conectividad que cuantifica la densidad de conexiones de un vértice hacia una comunidad disyunta, y su computación está basada en los valores de la similitud estructural dinámica. *ii)* Una definición de comunidad llamada Comunidad ϵ-Central que incrementa la probabilidad de detectar comunidades superpuestas preliminares en la estructura de comunidad disyunta. *iii)* Una función de probabilidad que computa la estructura de comunidad difusa a partir de la estructura de comunidad disyunta. Ya que este algoritmo presenta la misma complejidad computacional que el algoritmo original, entonces sigue siendo aplicable a redes complejas de gran escala. Finalmente, una experimentación extensiva ha sido desarrollada con el fin de probar las propiedades, eficacia y eficiencia de los algoritmos propuestos, y para compararlos con el estado del arte. Los resultados experimentales muestran que los algoritmos propuestos proveen un mejor balance entre calidad de la estructura de comunidad detectada, eficiencia de computación y facilidad de uso, comparados con el estado del arte.

Keywords: algorithm, complex networks, community detection, dynamic structural similarity, large-scale networks, graph clustering

Palabras clave: algoritmo, redes complejas, detección de comunidades, similitud estructural dinámica, redes de gran escala, agrupamiento en grafos

Acceptation Note

Thesis Work

" mention"

Jury
Luis Fernando Niño Vasquez

Jury
Jorge Eduardo Ortiz Triviño

Advisor
Elizabeth León Guzmán

Coadvisor
Jonatan Goméz Perdomo

Bogotá, D.C., October 2018

Dedication

To my parents, Marlene and Moisés. Thank you so much for supporting me in the pursuit of my dreams.

Acknowledgement

This thesis work would not be possible without the invaluable support from several people. First of all, I want to thank Castrillo-Velilla Family for their kindness and hospitality. I also thank my thesis advisor Elizabeth León Guzmán and co-advisor Jonatan Gómez Perdomo, who trusted and supported me during the course of the master's degree. Camilo Alaguna, and Rodrigo Moreno contributed with their reviews to improve the quality of this work; I thank them greatly.

Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

Networks are ubiquitous because they conform the backbones of many complex systems, such like social networks, protein-protein interactions networks, the physical Internet, the World Wide Web, among others [23]. In fact, a complex system can be modeled by a complex network, in such a way that the complex network represents an abstract model of the structure and interactions of the elements in the complex system[1] [51]. In real-world scenarios, these underlying complex networks tend to be large, with thousands and millions of elements and connections, change dinamically, have non-trivial topological features and present irregular patterns of connections among the elements. Moreover, these complex networks exhibit important structural properties such as *small-world*, *scale-free* and *community structure*. It could be possible to reveal valuable structural information and knowledge of how a complex system behave, if the structure and properties of its back-end complex network are discovered. For that reason, in the past two decades the study and analysis of complex networks and their properties has become an important research area in disciplines such as mathematics, biology, statistics, computer science, etc [52].

One important property of complex networks is the community structure. Detecting the community structure of a complex network is the task of grouping the set of elements (nodes) in such a way that elements in the same group are more similar to each other than to those in the other groups. This particular task is known as *Community Detection* and in this context, a group of nodes is denominated a *Community*. Intuitively, a community can be defined as a group of nodes where there are more connections (edges) inside the community than connections with the rest of the complex network [23].

Community Detection can be applied in many fields, like data mining, machine learning, pattern recognition, bioinformatics, social networks, among others, serving as a tool to make exploratory analysis, build predictive models and support decision making processes.

[1] For example, in social networks like Facebook, the network of user-user friendships, the social groups and the user reactions to posts, can be modeled as complex networks

For example, in applications that require social network analysis, community detection can be applied to find users with similar behaviors, to detect groups of interest, or to recommend products to final users in e-commerce platforms based on their shopping habits. To cite some examples, in the work developed by *Lalwani et al.*, [44] they employe community detection and collaborative filtering techniques to implement a recommendation system of movies, based on the analysis of Facebook graphs and movie rating datasets. Moreover, in the field of bioinformatics, we can see in the seminal work developed by *Spirin et al.* [66] how community detection is applied to detect groups of proteins that perform similar functions in the cell. Also, community detection has applications in cibersecurity to perform anomaly detection, for example, the application developed by *Wang et al.* in [67] aimed to detect the presence of a botnet and identifying the compromised nodes before the botnet becomes active.

Community Detection is not a specific algorithm, but the generic task to be solved, that is why in the past two decades several algorithms for community detection have been developed. Usually, the community detection algorithms differ in their conception of community or how efficiently find them. Moreover, the community detection algorithms can be categorized according to their functionality as hierarchical clustering, graph partitioning, partitional clustering, spectral methods, optimization techniques, dynamic methods, among others. A recent review with a detailed description and a comparative analysis of each category can be found in [27].

With the arrival of the Big Data era a vast amount of data is generated continuously. For example, many large-scale complex networks with thousands and millions of nodes and edges are generated from different sources such as social networks, peer-to-peer networks, the internet, etc [47, 62]. Algorithms are then required to handle data efficiently, so it is necessary to design and develop efficient methods to handle community detection on big networks, while keeping high quality results in terms of the detected community structure. However, algorithms for community detection with high quality results in small or medium sized networks are not suitable to work with large networks due to their high computational complexity (e.g., Agglomerative Hierarchical Walktrap [56], Divisive Hierarchical Girvan-Newman [28], Information Theory Infomap [63]). On the other hand, efficient algorithms in terms of computational complexity, compute community structure with some limitations, for example, the Louvain method [6] and its known problem of resolution limit [26], Label Propagation [58] and its convergence problems due to the random nature of its functioning, or Attractor [64] and the convergence problems presented in its underlying dynamic system, making it a slow algorithm in practice. Moreover, efficient algorithms that compute high quality community structure are not easy to tune in practice because they are too sensitive to the initial conditions given by the input parameters, for example, supervised algorithms that require the number of communities in advance as input parameter (this information is usually unknown in practice and is not easy to estimate) [48, 49], or algorithms that strongly depend on parameters that are not easy to tune in practice [72, 12]. So, the research efforts need to be focused to tackle the following challenges:

- Deal with large-scale complex networks through algorithms with low computational complexity.
- Find high quality and meaningful community structure.
- Build algorithms with intuitive input parameters and easy to tune in practice.

This thesis work try to answer the following question: How can we intuitively detect high quality community structure in large-scale complex networks?. To do so, we propose the following four contributions:

1. A novel similarity measure to characterize with high quality dense clusters in graphs. This new similarity computes the structural similarity of neighboring nodes based on the following recursive definition: *Two connected nodes are structurally similar if they share a structurally similar neighborhood.* Opposed to classical local structural similarities, our approach dynamically diffuses and captures information beyond the locality, describing the structural similarity of connected nodes in an entire graph at different points of time. This new similarity is modeled as an iterated function that can be solved by fixed point iteration in super-linear time and memory complexity, so it is able to analyze large-scale graphs.
2. The algorithm ISCAN is proposed as first approach to perform community detection in complex networks ISCAN. ISCAN is obtained after replacing the local structural similarity used in the algorithm SCAN with our dynamic structural similarity. Compared to the original SCAN algorithm, ISCAN improves the quality of the community structure and reduces its sensitivity to the parameter settings, while maintaining the same computational complexity.
3. A novel fast heuristic algorithm for multi-scale hierarchical community detection that can be applied efficiently to large-scale complex networks. The algorithm finds the community structure by clustering the nodes inspired on an agglomerative hierarchical algorithm composed of three steps. In the first step, a similarity measure for each edge is computed (several similarity measures can be adapted). In the second step, a heuristic builds meaningful communities as follows: each node is set in a separate community, then per iteration, each community C is merged with its most similar adjacent communities when C does not meet a specified community definition. This procedure iterates until all detected communities meet the specified community definition. In the third step, a heuristic merges communities as follows: per iteration, each community C is merged with its most similar adjacent communities when the C's size is less than a specified threshold. This procedure iterates until all detected communities achieve the minimum size required. Finally, the detected community structure is returned.
4. An extension of the previous algorithm that leverages the disjoint community structure to efficiently detect fuzzy and crisp overlapping community structure, while maintaining the same computational complexity.

1.1 Goal

The main goal of this thesis is to design and implement an algorithm for community detection in large-scale complex networks based on an agglomerative hierarchical clustering technique, unsupervised, without resolution limit, computationally efficient and competitive with the state-of-the-art. In order to achieve this goal, the following specific objectives were proposed:

1. To design and implement an agglomerative hierarchical algorithm with average time complexity of $O(|E|)$ in complex networks, unsupervised and without resolution limit.
2. To validate the algorithm empirically and statistically by using real-world and synthetic complex networks with ground-truth communities, and comparing it with the state-of-the-art.
3. To evaluate the performance of the algorithm empirically and statistically by applying it to real-world and synthetic large-scale complex networks, and comparing it with the state-of-the-art.

Methodology

The algorithms were developed using an iterative methodology in which each iteration consist of the following steps: algorithm design, algorithm codification, experiments, validations-evaluations and discussion. After each iteration, the detected problems in each step are tackled on the next iteration.

Moreover, this thesis work was written following a similar methodology and structure of a research article, aimed to produce in parallel research articles about the designed algorithms, experimental results and conclusions. Several articles were written in order to socialize this work on international academic scopes, more specifically, they were submitted for peer-reviewing to an international conference.

1.2 Contributions

The contributions of this thesis are mainly grouped into technical contributions and publications. These are listed below.

Technical Contributions

1. A novel dynamic structural similarity measure that characterize with high quality dense clusters in graphs. This structural similarity can be computed with average time complexity of $O(|E|)$ in complex networks.

2. An improved SCAN algorithm (ISCAN). Compared to the original SCAN algorithm, ISCAN improves the quality of the detected community structure and mitigates its sensitivity to the parameter settings, while maintaining the same computational complexity.

3. A novel heuristic algorithm for multi-scale and hierarchical community detection with average time complexity of $O(|E|)$ in complex networks, and competitive with the state-of-the-art.

   (a) An agglomerative hierarchical technique that instead of optimizing partition quality functions as usually does, builds communities by using per-cluster quality functions. This strategy avoids to build the entire dendogram in order to find a relevant hierarchy of communities.

   (b) A synchronous agglomerative hierarchical technique that does not require to sort the edges previous to the merge process.

4. A novel heuristic algorithm for fuzzy and crisp overlapping community detection with average time complexity of $O(|E|)$ in complex networks, and competitive with the state-of-the-art.

Publications

1. *Fast Heuristic Algorithm for Multi-scale Hierarchical Community Detection.* ASONAM '17 Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017, Pages 982-989. This paper won the *"Best Paper Award"* in the co-located symposium FAB '17.

2. *High-Quality Disjoint and Overlapping Community Structure in Large-Scale Complex Networks.* Accepted for presentation in the International Symposium on Foundations and Applications of Big Data Analytics (FAB) 2018.

1.3 Thesis Outline

This document is organized as follows:

- **Chapter 2** presents a literature review on the subjects the thesis deals with.

- **Chapter 3** describes the design and internals of the proposed dynamic structural similarity and shows an early application of it in the community detection task.
- **Chapter 4** describes the design and internals of the proposed community detection algorithm.
- **Chapter 5** develops experiments and discusses the results obtained with the proposed algorithms.
- **Chapter 6** draws some conclusions about this thesis and shows directions for future work.

CHAPTER 2

Literature Review

The goal of this chapter is to present a literature review on the topics that the thesis deals with. First, the theoretical notions of graphs, networks and complex networks used in this work are presented. Next, the relevant properties of complex networks related to this work are briefly reviewed. Then, a revision of the state-of-the-art centrality and similarity measures in graphs is discussed. Finally, the theoretical notions of community detection in complex networks and a review of the state-of-the-art algorithms in community detection are presented.

2.1 Graphs, Networks and Complex Networks

The solution found by Leonhard Euler to the Seven Bridges of Königsberg problem in 1736, laid to the foundations of graph theory (a field in discrete mathematics). Since then, graph theory has been fundamental to model and solve problems about the properties of artificial graphs. However, it has been adopted by other areas, such as biology, social sciences, computer science, physics, statistics in order to apply the graph theory to model and analyze networks that arise naturally in the real world [51]. Examples of naturally formed networks are the social networks, protein-protein interaction networks, as well as the World Wide Web and Telephone networks, that are information and communication networks respectively. The science of networks is the research field that studies both theoretical and real-world problems modeled by using networks.

In the past, the science of networks has allowed us to develop important breaktroughs in both social sciences and engineering applications, to cite for example the work developed by Granovetter with its "Strength of weak ties" [30] where he models human interactions, or the *PageRank* algorithm developed by Bring and Page [9] that supported the early versions of the Google[1] web search engine. Moreover, with the new advances in technology

[1] www.google.com

that bring better capacities in computing power and storage space, the science of networks goes beyond. Now it is possible to study in more detail the vast amount of data in form of networks of many natural and man-made complex systems. Analysing such complex systems is specially important to predict their functionality and understand their structural properties, organization and behaviors[52].

In the following subsections, the theoretical notions of graphs, networks and complex networks relevant to this thesis work are presented.

2.1.1 Graph Model

DEFINITION 1. Let $G = (V, E, \omega)$ be a graph with set of nodes V, set of edges $(u, v) \in E$ such that $u, v \in V$, and edge weighting function $\omega : E \mapsto \mathbb{R}$. In the case of undirected graphs, the edges (u, v) and (v, u) are considered the same. In the case of unweighted graphs $\omega(u, v) = 1$ for each $(u, v) \in E$. From now on, suppose G is an undirected and unweighted graph, unless other type of graph is explicitly mentioned.

DEFINITION 2. The structural neighborhood of a node u, denoted by $N(u)$, is defined as the open neighborhood of u; that is $N(u) = \{v \in V | (u, v) \in E\}$. Additionally, the closed structural neighborhood, denoted by $N[u]$, is defined as $N[u] = N(u) \cup \{u\}$.

DEFINITION 3. The degree of a node u, denoted by $d[u]$ and $d(u)$, is basically the cardinal of the structural neighborhood of u; that is $d[u] = |N[u]|$ and $d(u) = |N(u)|$ respectively.

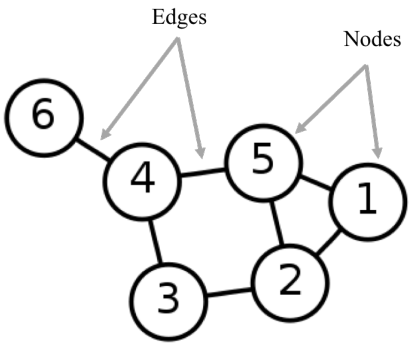


FIGURE 2.1. Unweighted undirected graph with 6 nodes and 7 edges. This image has been adapted. The original image is of public domain.

2.1.2 Network Model

A network is basically an abstract representation of a system, modeled conveniently by a graph that captures pair-wise element interactions or topological properties in the system. For example, in Figure 2.2 we can see an abstract formulation to the Seven Bridges of Königsberg problem that helped Euler to proof its solution. Furthermore, consider a social

network where the users and their mutual user-to-user relationships can be represented with nodes and edges respectively in an undirected graph, or a man-made system like the physical internet where a bunch of computers (nodes) are linked together by optical fiber (edges) and other data connections [52].

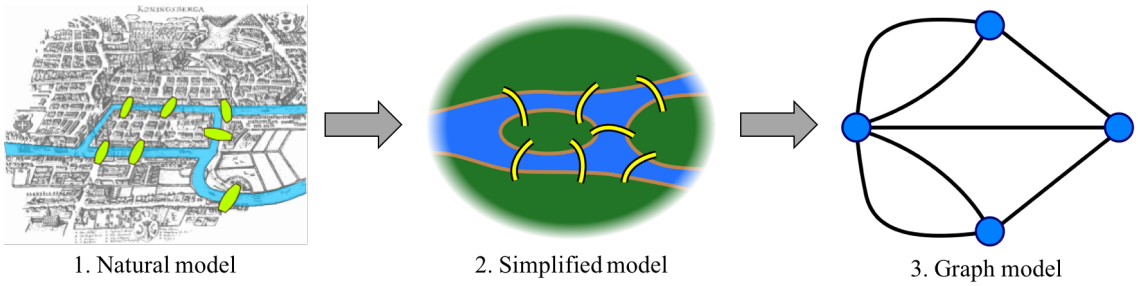


FIGURE 2.2. Graph formulation for the Seven Bridges of Königsberg problem. This image has been adapted. The original images are licensed under CC BY-SA 3.0.

2.1.3 Complex Network Model

Complex networks are just networks that conform the backbones of many complex systems. Because of their origins, complex networks present common characteristics and exhibit interesting topological properties. For example, nodes in complex networks are mostly separated by small geodesic distances[2] (also known as the "small-world" effect). Also, there are few nodes with high degree of connection[3] and many nodes with low degree of connection, so the degree distribution follows a power-law. Moreover, a complex network can be divided into clusters or communities that are groups of nodes with similar topological properties or specific patterns of connections, e.g., groups of nodes densely connected within the group and loosely connected with the rest of the network [23].

2.2 Properties of Complex Networks

2.2.1 Random Networks vs Complex Networks

Random networks are generative graph models based on random connectivity patterns between any pair of nodes in the graph. Random networks were first studied by Rapoport [59] and Erdős and Rényi [25] and both proposed a simple random graph model as follows: Let G be a graph with N nodes, where each pair of nodes in the G are linked together with random probability p (not linked with random probability $1-p$), so the node degree follows a Binomial Distribution. This graph model is known as the Erdős-Rényi graph model. Even though random graphs models, like the Erdős-Rényi model, are useful mathematical

[2]Geodesic distance is the shortest path (in number of edges) between two nodes.
[3]Degree of connection of a node is the number of edges connecting the node with the rest of the network.

models to study and proof the existence/absense of some topological properties in graphs [43, 24], they are barely present in real-world networks.

On the other hand, complex networks deviate from random networks mainly because of their generative models. While random networks are generated from fixed mathematical models, complex networks are the result of complex interactions guided by the (sometimes unpredictable) dynamics of the underlying system, for example, protein-protein interactions in the core of a cell or another living organism. For that reason, in complex networks the patterns of connections between elements are neither regular nor purely random [23].

2.2.2 Transitivity or Clustering

The Transitivity or Clustering of a networks is based on the following assumption: If a node X is directly connected[4] to node Y, and node Y is directly connected to node Z, then there is a high probability that node X will be directly connected to node Z, thus forming a triangular structure. The transitivity or clustering of a network means the existence of a high number of triangular structures in the network. There are two main coefficients to quantify the transitivity or clustering of a network: the *global clustering coefficient* [23] and the *local clustering coefficient* [68]. In general, the clustering coefficient measures the density of triangular structures in a network. In fact, this measure is expected to be higher in complex networks, compared to random networks with the same number of nodes and edges.

2.2.3 Degree Distribution

Suppose that p_k is the probability that a node in a network has degree k. In random networks, like the Erdős-Rényi model [25], any pair of nodes are linked together with probability p, thus the degree probability p_k follows a binomial distribution. In contrast, in complex networks, the probability p_k in the network strongly depends on the dynamics of the system, making the degree distribution unlikely to follow a binomial distribution. In fact, it has been shown empirically that complex networks tend to follow power-law distributions $p_k \sim k^{-\alpha}$ for some parameter α , or exponential distributions $p_k \sim e^{-k/\kappa}$ for some exponent κ [52]. Networks with power-law degree distributions are usually known as *scale-free* networks. Even though complex networks are scale-free, thare are several random graph models that generate scale-free networks, for instance, the model proposed by Barabási-Albert in [5], where each time a new node is added to the network it is linked to one or more existing nodes by preferential attachment.

[4]Two nodes are directly connected if they are linked together by an edge.

2.2.4 Maximum Degree

The maximum node degree k_{max} in a network could play an important role in some calculations, for that reason, it is important to have a good approximation of k_{max} given an arbitrary network. In complex networks, the value of k_{max} depends on the size of the network, in terms of the numbers of nodes n. If a complex network follows power-law degree distribution $p_k \sim k^{-\alpha}$, then a possible guess is that the maximum degree is near to the tail of the distribution. *Cohen et al.* [16] propose the value of $k_{max} \sim n^{1/(\alpha-1)}$ as an estimator of the maximum degree in a complex network.

2.2.5 Community Structure

Community Structure is the division of the nodes in a network into groups called *Communities* or *Clusters*, in such a way that nodes belonging to the same community are highly similar each other, but less similar with nodes outside the community [27]. What determines the similarity among nodes in a community depends on the particular problem of study. For example, in social networks like Facebook, it is very common to find community structure formed by group of users sharing common interests (e.g. hobbies, favorite sports, taste of music), also groups of people surrounding similar locations (e.g. hometown, university, library) or when they have similar occupations (e.g. co-workers, family members). In biological networks, like protein-protein interaction networks, communities are formed by proteins performing similar functions in the cell. Moreover, communities on IRC[5] may be shaped by groups of users that often discuss in the same channels.

2.2.5.1 Community Structure Model

DEFINITION 4. A *Community* is defined as a subset C of nodes in G, i.e., $C \subseteq V$. Also, a community can be defined as the induced subgraph $G[C]$ from the set of nodes in C.

DEFINITION 5. The *Community Structure* of a graph G, denoted by $C(G)$, is a set of communities extracted from G, i.e., $C(G) = \{C_i : C_i \subseteq V\}$. Moreover, each node must belong to at least one community, that is $\bigcup_{C_i \in C(G)} C_i = V$. The community structure can be classified into two main categories: disjoint and overlapping community structure.

DEFINITION 6. $C(G)$ is a disjoint community structure if $\bigcap_{C_i \in C(G)} C_i = \emptyset$, otherwise $C(G)$ is denominated an overlapping community structure. In other words, in the case of disjoint community structure, a node can belong to maximum one community, while in the overlapping community structure a node can belong to one or more communities.

DEFINITION 7. A *Fuzzy Overlapping Community Structure* $C^f(G)$, is an overlapping community structure in which each community $C_i \in C^f(G)$ is a fuzzy set (C_i, f_i), with

[5]Internet Relay Chat.

membership function $f_i : C_i \mapsto [0, 1]$. For each node $u \in C_i$, the value $f_i(u)$ is called the probability of membership of u in the community C_i.

DEFINITION 8. A *Crisp Overlapping Community Structure* $C^\alpha(G)$, is an overlapping community structure in which each community $C_i^\alpha \in C^\alpha(G)$ is a crisp set obtained with an α-cut of (C_i, f_i), i.e., $C_i^\alpha = \{u \in C_i : f_i(u) \geq \alpha\}$.

DEFINITION 9. The *Internal Degree* of connection of a node u in the community C, denoted $k_C^{int}(u)$ is the number of edges connecting the node u with nodes inside C.

DEFINITION 10. The *External Degree* of connection of a node u in the community C, denoted $k_C^{ext}(u)$ is the number of edges connecting the node u with nodes outside C.

2.2.5.2 Community Definitions

Several community definitions have been conveniently proposed from the general community definition (4), and each definition is aimed to be applicable in particular domains. Next, a general classification of the existing community definitions is briefly reviewed.

- **Intuitive Community Definition:** A community is defined as a group of nodes with high density of edges within the community and low density of edges toward the rest of network. For example, Figure 2.3 shows a small network with intuitive community structure.
- **Iternal Connectedness**: The communities are defined in terms of the number of connections between nodes in the community. The main variables are: The Internal degree k_C^{int}, the average internal degree $k_C^{avg-int}$ and the internal edge density δ_C^{int} [27].
- **External Connectedness**: The communities are defined in terms of the number of connections from the community towards the rest of the network. The main variables are: The external degree or cut k_C^{ext}, the average external degree or expasion $k_C^{avg-ext}$ and the external edge density or cut-ratio δ_C^{ext} [27].
- **Hybrid Connectedness**: The communities are defined as a combination of its internal and external connectedness. The main variables are: The total degree or volume k_C, the average degree k_C^{avg} and the Conductance C_C [27].
- **Classic View**: The classic community definitions can be further extended into two categories: Community definitions based on the the internal connectiviy patterns of the nodes in the community, for example, $n - cliques$, $n - clubs$, $n - clans$ and $k - plex$ [23]. Community definitions that take into account the relationship between the number of internal edges and external edges of a specified community. For example, the $Strong$ and $Weak$ communities defined in [57], and the $Strong$ and $Weak$ communities defined in [34].

DEFINITION 11. A community C is *Weak* in the sense of [57], if the total internal degree of C is greater than or equal to the total external degree of C, that is,

$$\sum_{u \in C} k_C^{int}(u) \geq \sum_{u \in C} k_C^{ext}(u) \tag{2.1}$$

DEFINITION 12. A community C is *Weak* in the sense of [34], if the total internal degree of C is greater than or equal to the total external degree of C towards each adjacent community, that is

$$\sum_{u \in C} k_C^{int}(u) \geq \max_{C_i \in C(G), C \neq C_i} |(u, v) \in E : u \in C \wedge v \in C_i| \tag{2.2}$$

From now on, we will refer to the *Weak* communities in the sense of [34] as *Most Weak* communities.

- **Modern View**: The community definition is based on the relationship between the internal probability of connections in the community and the external probability of connections towards the rest of the network. In other words, the probability of finding edges between nodes inside the community must be higher than the probability of finding edges towards the rest of the networks. In this category we find the popular Stochastic Block Model (SBM) that let us to defined more specialized communities, like assortative communities, disassortative communities and core-periphery communities, but also random graphs without community structure [27].

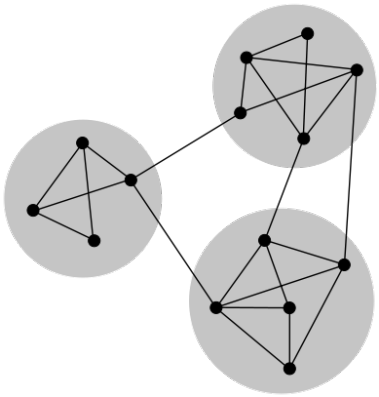


FIGURE 2.3. Small network displaying community structure with three communities (highlighted in gray) with high internal density of connections and low external density of connections. The original image is licensed under CC BY-SA 3.0.

2.3 Centrality and Similarity Measures in Graphs

It could be possible to predict the functionality or understand the behavior of a complex system if we get insights about it by analyzing its underlying network. For instance,

if we are capables of detecting groups of nodes with similar topological features in the network, we can get insights about the particular roles played by each node (e.g. hubs, outliers) or how entire groups (e.g. clusters) describe or affect the overall behavior of the complex system [23]. The main problem that arise in this kind of analyses is how to determine efficiently high quality topological or structural similarities in the network. For this reason, several methods have been proposed to try to cope the problem. For example, the local Degree centrality, the global Closeness centralitiy [23], Betweeness centrality[8] and Bridgeness centrality [37] that characterize the importance of a node or an edge in a network. Also, there are local structural similarity measures between nodes like Jaccard, Cosine or Dice that are based on the connectivity patterns of the nodes in their immediate neighborhood [12]. Additionally, we find more sofisticated methods like PageRank that ranks nodes in a network by using markov chain models [9], or SimRank that computes the similarity between nodes on directed graphs by using recursive similarity definitions [36].

In summary, a centrality, similarity or ranking measure is a function $f : V \mapsto \mathbb{R}$ or $f : E \mapsto \mathbb{R}$, such that nodes and edges respectively are mapped to real values to determine their importance[6] in the structure or functioning of the graph. A function f can be classified into two main categories: local and global. This classification depends on how much information is required from G in its computation. Usually, in global functions the prior knowledge of the entire graph is required, otherwise the functions are considered local.

In the following subsections, the state-of-the-art centrality and similarity measures related to this research work are discussed.

2.3.1 Centrality Measures

The degree centrality is based on the idea that nodes with high degree are involved more frequently in communications than nodes with low degree. For a node u, the degree centrality is basically its degree. This is a local function, because it only depends on the neighborhood of u. Several definitions based on the degree centrality have been proposed, like k-path centrality and edge-disjoint k-path centrality, and several randomnized algorithms have been proposed to compute them efficiently. [23]

The betweeness centrality is frequently used to measure the importance of nodes and edges in a graph. This centrality is based on the idea that the importance of a node/edge is proportional to the number of shortest paths that pass through the investigated node/edge. The higher the betweenes centrality, the more important are nodes/edges for communication pourposes [23]. By definition this is a global function. So, the main limitation of the betweeness centrality is its computational cost, since it requires $O(|E|)$ computations per node/edge and $O(|V||E|)$ for the entire graph, making it impractical for very large graphs [8, 37]. For that reason, several algorithms have been proposed to compute efficiently

[6] the definition of importance depends on the specific problem being studied.

approximate values. Those algorithms are usually based on sampling methods or (δ, ϵ) approximations [4, 61].

2.3.2 Dynamic Measures

The core idea behind dynamic measures is the concept of random walk. A random walk is an iterative process that starts from a random node, and at each step, either follows a random outgoing edge from the current node or jumps to a random node. PageRank (PR) [9] is a node ranking method that defines the importance of a node recursively as follows: *The importance of a node in the network is proportional to the importance of the nodes pointing to it.* This algorithm models a random surfer who is placed in a specific web page and then navigates the web by clicking on links. However, the surfer starts to navigate from a random web page with a probability given by a damping factor α (tuned by hand). The PR is modeled as the stationary distribution of a markov chain process solved by fixed point iteration. Several dynamical systems have been proposed since the original PR, like Personalized PageRank (PPR) [49], Heat Kernels (HK) [41] and pure random walks (RW) [56]. All of them compute similarities for seed nodes respect to the whole graph, hindering the simultaneously computation of multiple similarities.

Another popular dynamic similarity based on the random walk intuition is SimRank [36] and others derived from it [39]. SimRank defines structural-context similarity of nodes (directly connected by edges or not) recursively as follows: *Two objects are similar if they are related to similar objects.* The SimRank is modeled as a recursive function solved by fixed point iteration. By definition, SimRank only works for directed graphs, and also requires a decay factor C in order to control the flow of information in the dynamic system and to achieve convergence.

An algorithm to compute structural similarity based on distance dynamics have been proposed in [64]. They propose a fast algorithm to detect high-quality community structure by merging nodes with high structural similarity. The structural similarity is defined as the result of three interaction patterns that dynamically change the similarity through the time. These interaction patterns are solved by fixed point iteration. Although the system presents dynamic behavior, they force its convergence by truncating similarities above one, and below zero.

2.3.3 Structural Similarity

DEFINITION 13. ***Structural Equivalence***: Two nodes in a network are structurally equivalent if they share the same neighborhood; that is, given two nodes u and v, then $N(u) = N(v)$.

However, computing $|N(u) \cap N(v)|$ is not considered a good similarity measure by itself, because it has not into account the degrees of the nodes. The structural equivalence can be improved by normalizing its value as the *Structural Similarity* does.

DEFINITION 14. ***Structural Similarity (a.k.a Cosine Similarity)***: The local structural similarity (LSS) of nodes u and v, denoted by $\sigma(u, v)$, is defined as the cardinal of the set of common neighbors $|N[u] \cap N[v]|$, normalized by the geometric mean of their degrees [72], that is,

$$\sigma(u, v) = \frac{|N[u] \cap N[v]|}{\sqrt{d[u] \times d[v]}} \tag{2.3}$$

By definition, the structural similarity is a local function because only requires information about the inmediate neighborhood of the nodes u and v. In fact, given two nodes u and v, the structural similairy $\sigma(u, v)$ can be computed in $O(min(d[u], d[v]))$ time. Furthermore, $O(\alpha(G) \times |E|)$ time is required to compute the structural similarity for each pair of nodes in a graph G, where the term $\alpha(G)$ corresponds to the arboricity of G [14]. The structural similarity is just an extension to the context of graphs of the Cosine Similarity. Another two popular similarity measures extended to the context of graphs are Jaccard and Dice, defined as $J(u, v) = |N[u] \cap N[v]| \div |N[u] \cup N[v]|$ and $D(u, v) = 2 \times |N[u] \cap N[v]| \div (d[u] + d[v])$ respectively.

2.4 Community Detection in Complex Networks

Community Detection is the general task of finding the underlying community structure in a complex network. The community structure can help us to understand the topological structure and behavior of a complex network, and therefore get insights about the complex system being studied. For that reason, community detection has become an attractive research topic the past few years. Moreover, community detection can be used in engineering applications, due to it serves as base technology to develop computer programs aimed to solve real-world problems (As described in Chapter 1). Maybe, to perform community detection in small networks can be carried with easy, but in the context of complex networks it becomes a non-tivial task, due to complex networks are large, with thousands and millions of nodes and edges. For that reason, detecting the community structure is both a challenging and computational expensive task, thus efficient computational tools (data structures and algorithms) are required to overcome with the challenge.

In summary, Community Detection consists in finding a community structure (5) in a graph G. By definition, community detection is an instance of graph clustering, so it is classified as NP-Hard problem. If the number of communities to detect is specified in advance as parameter of the algorithm, then the community detection is considered supervised, otherwise it is considered unsupervised.

2.4.1 State-of-the-art Algorithms in Community Detection

Several algorithms for community detection have been developed in order to perform community detection. Usually, the community detection algorithms differ in their conception of community (See 2.2.5.2) and their computational complexity. Moreover, the community detection algorithms can be categorized according to their functionality as hierarchical clustering, graph partitioning, partitional clustering, spectral methods, optimization techniques, dynamic methods, among others [27].

This thesis work tries to deal with unsupervised community detection in large-scale complex networks, for that reason, a special emphasis is made on the state-of-the-art unsupervised algorithms that can be efficiently applied in large-scale scenarios. An algorithm is considered unsupervised, if the number of communities to detect is not required as input parameter. Moreover, a particular algorithm can be considered efficient, if its time complexity is linear in terms of the number of edges in the input graph, i.e., time complexity of $O(|E|)$ in the worst or average case.

Next, a general classification of the state-of-the-art algorithms for disjoint and overlapping community detection is briefly presented.

Optimization Techniques: The idea behind these methods is that a good community structure must present high values of the Modularity score [53]. Fast Greedy [15] performs greedy modularity optimization with an agglomerative hierarchical approach. Multilevel [6] is a fast modularity optimization algorithm that performs an agglomerative hierarchical approach composed of two phases: network collapse and greedy optimization. These two algorithms find good local optima of the modularity. However, it has been proved that optimizing the modularity yields to the problem of resolution limit, making the modularity-based methods unable to detect communities smaller than a certain size that depends on the size of the network [26]. Infomap [63] applies a greedy technique to minimize an objective function called the map equation. The map equation quantifies the information needed to represent a random walker in a network using a two-level nomenclature. Infomap can detect both, disjoint and overlapping community structure. Experimentally, Infomap has shown high computational complexity in large-scale complex networks [10]. Several multi-resolution modularity definitions have been proposed, but they tend to divide big communities to favor the small communities, that means, the multi-resolution modularity definitions also present resolution limit [27]. In fact, any algorithm that optimizes partition quality functions, like modularity, will yield to some resolution limit [26, 40].

Information Propagation: Label Propagation proposed by Raghavan et al. [58] simulates the diffusion of information (labels) through the network. At the beginning, each node is labeled with a unique value, then iteratively each node takes the most frequent label in its neighborhood and the process continues until convergence. The results provided by Label Propagation are sometimes unpredictable and the whole network can be detected as a single community. Experimentally, Label Propagation requires a large number of it-

erations until convergence, thus it can take long running times on large-scale networks. Another algorithm based on the label propagation technique is the Speaker-Listener Label Propagation (SLPA) [71]. SLPA is a linear time algorithm to detect disjoint and overlapping community structure in complex networks, employing a general speaker-listener information propagation process. Experimentally, SLPA presents the same variability of LP in the resulting community structure for different runs over the same network. Recently, a fast algorithm was proposed in [38]. It uses a two-steps method to build the community structure based on label propagation. In the first step, the structural similarity of nodes is computed for each pair of adjacent nodes. Additionally, it applies the label propagation algorithm to detect meta-communities conformed of most similar nodes. In the second step, a multilevel label propagation technique is applied to build communities that meet a modified version of the Weak community definition [57]. The second step is based on two sub-steps: network collapse and label propagation. The algorithm introduces a cohesion parameter α (a real value within the range [0, 1]) to control the resolution of the resulting communities. The Community Overlap PRopagation Algorithm (COPRA) [31] is an adaptation of the original Label Propagation algorithm to detected overlapping communities. COPRA performs the same propagation algorithm, but allows each node to belong to more than one community. Fluid Communities [54] is a recent algorithm based on label propagation technique, however this algorithm is supervised, because it requires the number of communities as parameter. A Local-First Discovery Method for Overlapping Communities [20] (DEMON) allows nodes to vote for local communities in their ego-neighborhood using a label propagation technique, then the local communities are merged into a global collection of communities.

Structural Clustering: The SCAN [72] algorithm and its variants, pSCAN [12], SCAN++ [65], Index-Based SCAN [69] cluster dense zones of nodes determined by the structural similarity of nodes. They are fast but strongly depend on a minimum similarity parameter ε that is difficult to estimate. To overcome the problem of estimating ε, the parameter-less algorithms SHRINK-H and SHRINK-G [35] were proposed. SHRINK-H performs agglomerative hierarchical clustering by merging dense pairs into micro-communities that conforms the hierarchy of communities, and the final clustering is determined by the partition that maximize the modularity. SHRINK-H presents a worst-case time complexity making it unable to handle efficiently large datasets. SHRINK-G is faster but it sacrifices the capacity of finding hierarchical community structure. AHSCAN [74] performs agglomerative hierarchical clustering by iteratively merging pair of nodes in order of decreasing structural similarity of nodes, until a single cluster remains. AHSCAN selects the partition that maximizes the modularity, and its time complexity scales by $O(|E| \times |V|)$.

Random Walks: WalkScan [33], HeatKernels [41], PersonalizedPageRank [49] and LEMON [48] are recently proposed algorithms to compute local community structure based on the simulation of random-walk processes from seed nodes 2.3.2. The main limitation of those algorithms is the lack of clear methodology to choose the seed nodes, also the number of detected communities strongly depends on the number of selected seed

nodes, making them basically supervised algorithms for community detection. Another popular algorithm is the Markov Cluster Algorithm (MCL) [22]. MCL simulates flow of information in a graph by using a stochastic process composed of two steps: inflation and contraction. The intuition behind MCL is that there is a high flow of information inside dense clusters, but the information rapidly evaporates when it flows between sparsely connected clusters.

Others: Other recently proposed algorithm is Attractor [64]. Attractor investigates local distance dynamics among connected nodes. Attractor computes the distance on edges based on the Jaccard similarity and applies 3 interaction patterns until the distances converge. The resulting communities are the connected components generated through the removal of the edges with final distance of one. Attractor introduces a cohesion parameter α (a real value within the range [0, 1]) to control the resolution of the detected communities. Greedy Clique Expansion (GCE) [46] is an efficient algorithm to perform overlapping community detection by expanding seed communities in order to optimize a fitness function. Those seed communities are the maximal cliques in the graph that are found automatically by the algorithm.

2.5 Summary

This chapter presented a literature review on the topics that the thesis deals with. First, the theoretical notions of graphs, networks and complex networks were presented. Next, the relevant properties of complex networks related to this work were briefly discussed. Then, the state-of-the-art centrality and similarity measures were reviewed. Finally, the theoretical notions of community detection in complex networks and the state-of-the-art algorithms in community detection were presented.

CHAPTER 3

Dynamic Structural Similarity

The local structural similarity $\sigma(u, v)$ (See 2.3.3) is a good quality measure, but it is limited to the immediate neighborhood of u and v. This limitation bypass important structural properties given by patterns of connections beyond the locality (paths of length 2, 3, 4, ..., N between u and v). One approach to solve the aforementioned limitation is by computing explicitly paths with length greater than one between the nodes u and v. This computation is done by performing complete enumeration, like the local edge clustering coefficient proposed in [57]. This coefficient is defined for the edge (u, v) as the number of cyclic structures of length k that the edge (u, v) belongs to, normalized by the total number of cyclic structures of length k that can be build given the degrees of the nodes u and v. Other similarity measures based on complete enumeration are the global closeness and betweeness centralities [23], that count the number of all-pairs shortest paths running through the edge (u, v). The disadvantage of these approaches is the high computational complexity required to enumerate the paths, making them impractical in large-scale graphs.

In this chapter, a novel structural similarity measure is proposed. Our proposal determines the structural similarity of connected nodes in a graph by dynamically diffusing and capturing information beyond the immediate neighborhood of the analyzed nodes, and can be used to analyze large-scale graphs, due to it can be computed efficiently.

This chapter is organized as follows: Section 3.1 describes the design and internals of the proposed Dynamic Structural Similarity, followed by the reference algorithm to compute it efficiently (3.1.1), and its corresponding complexity analysis (3.1.2). Section 3.2 presents a first application of the Dynamic Structural Similarity in the community detection task with the proposed ISCAN algorithm.

3.1 Dynamic Structural Similarity Definition

In order to compute structural similarity without doing complete enumeration, we propose a diffusion system to spread and capture structural similarity. Our approach is based on the following intuition: *Two nodes are structurally similar if they share a structurally similar neighborhood.* Let $DSS(u, v)$ be the *Dynamic Structural Similarity* (DSS) of u and v. Following the intuitive definition a recursive function $DSS(u, v)$ is as follows,

$$DSS(u,v) = \frac{\sum_{x \in N[u] \cap N[v]} DSS(u,x) + DSS(v,x)}{\sqrt{\sum_{x \in N(u)} DSS(u,x) \times \sum_{y \in N(v)} DSS(v,y)}} \tag{3.1}$$

Following the idea of the structural equivalence (13), we extend the similarity of two nodes u and v, not only to the cardinal of their common neighborhood, but to their common similarity, i.e., the sum of the similarities of the edges connecting u and v to each of their common neighbors x. The higher the total similarity in the common neighborhood, the more structurally similar must be the nodes u and v. The common similarity correspond to the numerator in the Equation 3.1. Additionally, the common similarity is divided by the geometric mean of the total similarity in the neighborhood of u and the total similarity in the neighborhood of v. Such division is done in order to get the relative importance of the common similarity respect to the entire neighborhood (denominator), similar to the normalization performed in the local structural similarity (Equation 2.3).

Under this similarity definition, if the edge $(u, u) \in E$, then the node u will present a similarity of 2 respect to itself, i.e., $DSS(u, u) = 2$. Also, from Equation 3.1 is easy to see that DSS is symmetric, i.e., $DSS(u, v) = DSS(v, u)$. Although the structural equivalence is defined for any pair of nodes, we set as base case $DSS(u, v) = 0$ if $(u, v) \notin E$ (See section 3.1.2 for the explanation of this constraint).

Intuitively, the term in the numerator contribute positively to $DSS(u, v)$, this contribution is directly proportional to the total similarity in the common neighborhood of u and v. On the other hand, the term in the denominator contributes negatively to $DSS(u, v)$, this contribution is inversely proportional to the geometric mean of the total similarity in the neighborhood of u and the total similarity in the neighborhood of v. In fact, the main negative contribution is given by the nodes that are not common neighbors of u and v.

3.1.1 Computing the Dynamic Structural Similarity

The DSS in a graph G can be computed by fixed point iteration. Let DSS_t be the iterated function defined from DSS on iteration t. For each iteration t, $|V|^2$ entries $DSS_t(u, v)$ are maintained, $DSS_t(u, v)$ gives the similarity measure between nodes u and v on iteration t. The next iteration $DSS_{t+1}(u, v)$ is computed based on $DSS_t(u, v)$. In the initialization

step (when $t = 0$), if the graph is un-weighted, the value of each $DSS_0(u, v)$ must be assigned to an equal and positive similarity score s (any real positive value), and zero if the edge does not exists in G.

$$DSS_0(u, v) = \begin{cases} s & \text{if } (u, v) \in E \\ 0 & \text{if } (u, v) \notin E \end{cases} \tag{3.2}$$

In the case of weighted graphs, the initial similarity can be set to the edge's weight, i.e., $DSS_0(u, v) = \omega(u, v)$.

To compute $DSS_{t+1}(u, v)$ from $DSS_t(u, v)$, the Equation 3.1 is adapted as follows,

$$DSS_{t+1}(u, v) = \frac{\sum\limits_{x \in N[u] \cap N[v]} DSS_t(u, x) + DSS_t(v, x)}{\sqrt{\sum\limits_{x \in N(u)} DSS_t(u, x) \times \sum\limits_{y \in N(v)} DSS_t(v, y)}} \tag{3.3}$$

3.1.2 Complexity Analysis

Let $E_s \subseteq |V|^2$ be the set of node pairs to whose DSS is being computed. Setting DSS_0 in the initialization step takes $O(|E_s|)$ time. Furthermore, if the adjacent list for each node is sorted, then $N[u] \cap N[v]$ can be computed in $O(min(d[u], d[v]))$ time. Thus, one iteration of the iterated structural similarity (Equation 3.3) requires $\sum_{(u,v) \in E_s} min(d[u], d[v])$ operations. In [14] it has been proved that the number of operations performed per iteration in Equation 3.3 has an upper-bound of $2 \times \alpha(G) \times |E_s|$, such that $\alpha(G) \leq \sqrt{|E|}$ is the arboricity of G. Finally, T iterations of Equation 3.3 are performed, resulting in a total complexity of $O(T \times \alpha(G) \times |E_s|)$ time. As opposed to other dynamic similarity measures [39], we set $DSS(u, v) = 0$ whenever $(u, v) \notin E$ in order to reduce the number of similarity computations from a maximum of $O(|V|^2)$ entries to $O(|E|)$ entries. This reduction becomes specially important on sparse graphs with $|E| = O(|V|)$. The space complexity is $O(|V| + |E|)$.

3.2 ISCAN - Improved SCAN Algorithm

The structural similarity has been employed as a measure of cohesion within clusters and becomes specially useful to approximate dense subgraphs in networks with high *Transitivity* and *Community Structure* [51], so it has been employed to perform community detection in complex networks [10, 64, 12, 72, 13, 38]. SCAN [72] is an algorithm that takes full advantage of the local structural similarity to perform community detection. SCAN is based on the idea of structure connected clusters expanded from seed core nodes. The core nodes are nodes with a minimum of μ adjacent nodes with structural similarity that

exceeds a threshold ϵ [72]. A structure connected cluster $C_{\mu,\epsilon}$ is a maximal subset of nodes in which every node in C is structure reachable from some core node in C. The SCAN algorithm requires two parameters: the minimum number of points μ and the minimum accepted similarity ϵ, with default values of 2 and [0.5, 0.7] respectively according to the authors. Moreover, SCAN is one of the fastest algorithm in the literature, with total time complexity of $O(\alpha(G) \times |E|)$.

Because the SCAN algorithm performs quite well in terms of the quality of the resulting clustering, the majority of research works based on it, are focused to improve SCAN in terms of its computational complexity but are not focused in the quality of the detected community structure or its usability. For example, the three recently proposed algorithms pSCAN [12], SCAN++ [65] and Index-Based SCAN [69] compute the same clustering results with optimized time complexities. In contrast, this research work is focused to improve the quality of the detected community structure and to reduce the sensitivity to the parameter settings of SCAN, while maintaining the same computational complexity in practical cases.

We consider that the key ingredient to improve the quality of the results and to reduce the sensitivity of SCAN, is to support the cluster construction on a robust similarity measure capable of differentiating with higher quality inter-cluster edges from intra-cluster edges. For that reason, the ISCAN algorithm is proposed. ISCAN is obtained after replacing the local structural similarity used in SCAN with the proposed Dynamic Structural Similarity. ISCAN presents a time complexity of $O(T \times \alpha(G) \times |E|)$, given T as the the number of iterations performed by the back-end DSS. Because T does not scale with the size of network ($T \approx 5$, See Chapter 5), it can be considered a constant factor in most practical cases, therefore we argue that ISCAN keeps the same asymptotic time complexity of the original SCAN algorithm.

Experiments

The experimental evaluation and validation of the Dynamic Structural Similarity and the ISCAN algorithm is performed in Section 5.2.

3.3 Summary

In this chapter, a novel Dynamic Structural Similarity measure was proposed. This similarity is aimed to overcome the limitations presented by the local structural similarity measures, and the prohibitive computational complexity presented by the similarities that perform complete enumeration. This new similarity is modeled as an iterated function that can be solved in super-linear time complexity in terms of the size of the input, so it can be efficiently applied to large-scale graphs. Also, the ISCAN algorithm was presented as a modification of the SCAN algorithm. ISCAN is obtained after replacing the

local structural similarity used in SCAN with the proposed Dynamic Structural Similarity. Compared to the original SCAN algorithm, ISCAN is expected to improve the quality of the detected community structure and reduce its sensitivity to the parameter settings.

CHAPTER 4

Community Detection Algorithm

Many algorithms have been developed to perform community detection based on different approaches (See Section 2.4.1). But the approach based on agglomerative and/or hierarchical techniques has shown a good trade-off in terms of quality of the detected community structure and speed of computation. For that reason, we propose a novel fast heuristic algorithm for multi-scale hierarchical community detection inspired on an agglomerative hierarchical clustering technique.

This chapter presents a novel agglomerative hierarchical algorithm that does not merge only clusters with maximal similarity as in the classical approach, but merges any cluster that does not meet a specified community definition with its most similar adjacent clusters. The algorithm computes all the similar clusters at the same time is checking if each cluster meets the specified community definition. It is done in linear time complexity in terms of the number of cluster in the iteration. Since a complex network is a sparse graph, this approach has a super linear time complexity with respect to the size of the input, making it suitable to be applied on large-scale complex networks.

This chapter is organized as follows: Section 4.1 describes the design and internals of the proposed algorithm. Section 4.2 shows a synchronous implementation of the proposed algorithm and its corresponding complexity analysis. In Section 4.3, an extension of the algorithm is proposed in order to detect overlapping communities.

4.1 Heuristic Agglomerative Hierarchical Clustering

The heuristic algorithm for multi-scale and hierarchical community detection proposed in this chapter is based on the following assumptions:

- **Assumption 1**: If a node u presents a maximal similarity measure with an adjacent node v, then u is more likely to be in the same community as v.

- **Assumption 2**: If a community C_1 presents a maximal similarity measure with an adjacent community C_2 through an edge directly connecting C_1 and C_2, then C_1 and C_2 are more likely to be part of the same community in the next hierarchical level. Whether to merge C_1 to C_2 depends exclusively of the current state of C_1.

Based on the two assumptions, the proposed algorithm finds the community structure inspired on an Agglomerative Hierarchical Clustering technique (AHC) [23] composed of three steps.

- **Step 1**: The structural similarity for each edge is computed once by using Equation 3.3.
- **Step 2**: The first hierarchical level is computed as follows: each node is set in a separate community, then by assumptions 1 and 2, per iteration, each community C is merged with its most similar adjacent communities, when C does not meet a community definition passed by parameter (instead of merging only the communities with maximal/minimal similarity in the iteration, as in the classical AHC). This procedure iterates until all detected communities meet the community definition (instead of iterating until a single community remains, as in the classical AHC).
- **Step 3**: Similarly to Step 2, a next hierarchical level is computed as follows: by assumption 2, each community C is merged with its most similar adjacent communities, when the C's size is less than a threshold passed by parameter. This procedure iterates until all detected communities achieve the minimum size required.

The proposed algorithm can be adapted to any similarity measure capable of describing dense sub-graphs (e.g. Cosine, Jaccard, Dice. See Section 3.2). In fact, in early version of the proposed algorithm, the Cosine similarity was selected by default. However, it has been replaced it with the Dynamic Structural Similarity (DSS) proposed in Chapter 3, since the DSS increases the probability of identifying intra-cluster edges and inter-cluster edges.

The community definitions Weak 2.1 and Most Weak 2.2 have been selected as fitness functions to determine the cut in the dendogram that represents a first hierarchy of communities. Those community definitions have been selected for the following reasons: *i*) If a random network (e.g., Erdős-Rényi model) is divided into two disjoint groups of nodes, there is a low probability that the two groups fulfills the community definition [57]; *ii)* the Weak and Most Weak definitions are local and independent of the size of the network. An appropriate local heuristic that uses the selected community definitions as cluster quality functions, instead of partition quality functions, can build communities without resolution limit; *iii)* the Weak and Most Weak definitions are compatible with other community definitions. For example, a community defined as Strong [57] can be also defined as Weak, while the opposite is not always true. The same property applies to other Strong and Weak definitions [34]. *iv)* The Weak definition can be used to detected coarse-grained

community structure, and the Most Weak definition can be used to detected fine-grained community structure.

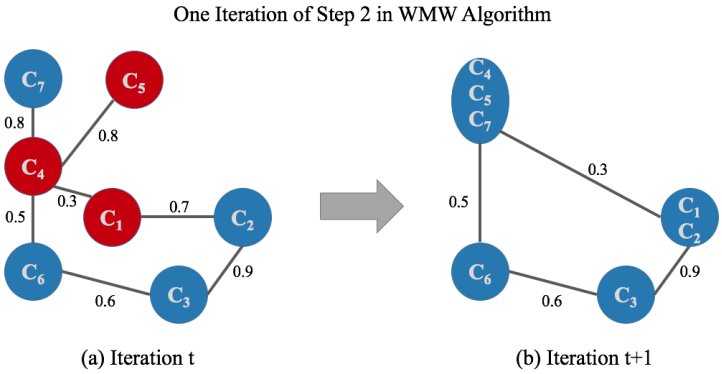


FIGURE 4.1. Example of one iteration of Step 2 in the proposed algorithm. At iteration t the network is composed of seven communities and at iteration $t+1$ the network is composed of four communities. Communities in red do not meet the community definition, while communities in blue meet the community definition. Only the edges with maximum similarity between two communities are displayed.

Figure 4.1 shows a generic example of one iteration of Step 2 in the proposed algorithm (Step 3 is performed in a similar way). As we can see, per iteration, more than one community can be merged, allowing faster convergence compared to the classical AHC technique. Also, the algorithms stops at iteration $t+1$ since all the detected communities meet the community definition.

4.2 Synchronous Implementation of the Algorithm

The proposed heuristic agglomerative hierarchical clustering algorithm can be implemented synchronously and asynchronously. For the purpose of this work, we provide a detailed pseudo-code of a synchronous implementation (See Algorithms 1, 2, 3, 4). From now on, we will refer to the proposed algorithm as the WMW algorithm (acronym of Weak and Most Weak community detection).

For this implementation, we assume without loss of generality that nodes in G are numbered from zero to $|V|-1$; otherwise, the vertice's identifiers must be normalized before applying the algorithm. Additionally, the community structure is represented using a disjoint-set data structure [19] in order to perform the following operations optimally: node's community querying and adjacent communities merging. This community representation was inspired in the way the gravitational clustering algorithm developed by Gomez et al. [29] represents clusters on spatial data.

WMW Algorithm (See Algorithm 1): It is the main of the algorithm. From line 1 to 2, the Dynamic Structural similarity for each edge is computed. At line 4 each node is assigned into a separate community. From line 5 to 9 the community structure with

Algorithm 1 WMW

Require: $G = (V, E)$, $minSize$, CD, T
Ensure: Disjoint Community Structure C
1: **for all** $(u, v) \in E$ **do**
2: Compute $\alpha(u, v) = DSS_T(u, v)$
3: **end for**
4: $C \leftarrow$ Disjoint-set of size $|V|$ // C_i = Community of node i-th
5: **if** $CD = WEAK$ **then**
6: WeakCommunityDetection(G, C, α)
7: **else**
8: MostWeakCommunityDetection(G, C, α)
9: **end if**
10: MinimumCommunitySize(G, C, α, K)

the required community definition CD is computed, and in line 10 the hierarchical level is selected.

Weak Community Detection (See Algorithm 2): Builds a community structure where each community meets the Weak community definition. The array position W_C stores the maximal structural similarity found so far, that connects the community C to the adjacent communities stored in the vector R_C. The array position B_C stores the difference between the internal and external degree of the community C. In the case $B_C < 0$, then the community C does not meet the Weak definition. From line 31 to 38, each community C that does not meet the Weak definition, is merged with its most similar adjacent communities.

Most Weak Community Detection (See Algorithm 3): Builds a community structure where each community meets the Most Weak community definition. The array position W_C stores the maximal structural similarity found so far, that connects the community C to the adjacent communities stored in the vector R_C. The array position B_C stores the internal degree of the community C and the array position D_C stores the maximal external degree of the community C towards an adjacent community. The key $H_{i,j}$ (and $H_{j,i}$) stores the external degree between adjacent communities i and j. If $B_C < D_C$, then the community C does not meet the Most Weak definition. From line 33 to 39, each community C that does not meet the Most Weak definition, is merged with its most similar adjacent communities.

Minimum Community Size Detection (See Algorithm 4): Builds communities having a size no less than K. The array W and the hash table R have the same meaning of that in Algorithms 2, 3. Additionally, the array position S_C stores the current size of the community C. From line 27 to 34, each community C whose current size S_C is less than K, is merged with its most similar adjacent communities.

Algorithm 2 WeakCommunityDetection

Require: $G = (V, E)$, C, α
Ensure: Disjoint Community Structure C

let W, B be two arrays of size $|V|$
let R be a hash table
let $flag =$ **true**
while $flag =$ **true do**
    $flag \leftarrow$ **false**
    for all $c \in C$ **do**
        $W_c = -1$, $B_c = 0$, $R[c].clear()$
    end for
    for all $(u, v) \in E$ **do**
        if $C_u \neq C_v$ **then**
            $B_{C_u} = B_{C_u} - 1$
            $B_{C_v} = B_{C_v} - 1$
            if $\alpha(u, v) > W_{C_u}$ **then**
                $W_{C_u} = \alpha(u, v)$
                $R[C_u].clear()$
            end if
            if $\alpha(u, v) = W_{C_u}$ **then**
                $R[C_u].add(C_v)$
            end if
            if $\alpha(u, v) > W_{C_v}$ **then**
                $W_{C_v} = \alpha(u, v)$
                $R[C_v].clear()$
            end if
            if $\alpha(u, v) = W_{C_v}$ **then**
                $R[C_v].add(C_u)$
            end if
        else
            $B_{C_u} = B_{C_u} + 2$
        end if
    end for
    for all $c \in C$ **do**
        if $B_c < 0$ **and** $R[c].size() > 0$ **then**
            for all $x \in R[u]$ **do**
                $C.merge(c, x)$
            end for
            $flag \leftarrow$ **true**
        end if
    end for
end while

Algorithm 3 MostWeakCommunityDetection

Require: $G = (V, E)$, C, α
Ensure: Disjoint Community Structure C
let W, B, D be three arrays of size $|V|$
let R, H be two hash tables
let $flag =$ **true**
while $flag =$ **true do**
    $flag \leftarrow$ **false**
    for all $c \in C$ **do**
        $W_c = -1$, $B_c = 0$, $D_c = 0$, $R[c].clear()$
    end for
    for all $(u, v) \in E$ **do**
        if $C_u \neq C_v$ **then**
            $H_{C_u,C_v} = H_{C_u,C_v} + 1$
            if $H_{C_u,C_v} > D_{C_u}$ **then**
                $D_{C_u} = H_{C_u,C_v}$
            end if
            if $H_{C_u,C_v} > D_{C_v}$ **then**
                $D_{C_v} = H_{C_u,C_v}$
            end if
            if $\alpha(u, v) = W_{C_u}$ **then**
                $R[C_u].add(C_v)$
            end if
            if $\alpha(u, v) > W_{C_v}$ **then**
                $W_{C_v} = \alpha(u, v)$
                $R[C_v].clear()$
            end if
            if $\alpha(u, v) = W_{C_v}$ **then**
                $R[C_v].add(C_u)$
            end if
        else
            $B_{C_u} = B_{C_u} + 2$
        end if
    end for
    for all $c \in C$ **do**
        if $B_c < D_c$ **and** $R[c].size() > 0$ **then**
            for all $x \in R[u]$ **do**
                $C.merge(c, x)$
            end for
            $flag \leftarrow$ **true**
        end if
    end for
end while

Algorithm 4 MinimumCommunitySize

Require: $G = (V, E)$, C, α, K
Ensure: Disjoint Community Structure C

let W, S be two arrays of size $|V|$
let R be a hash table
let $flag =$ **true**
while $flag =$ **true do**
    $flag \leftarrow$ **false**
    for all $c \in C$ **do**
        $W_c = -1$, $S_c = C.size(c)$, $R[c].clear()$
    end for
    for all $(u, v) \in E$ **do**
        if $C_u \neq C_v$ **then**
            if $\alpha(u, v) > W_{C_u}$ **then**
                $W_{C_u} = \alpha(u, v)$
                $R[C_u].clear()$
            end if
            if $\alpha(u, v) = W_{C_u}$ **then**
                $R[C_u].add(C_v)$
            end if
            if $\alpha(u, v) > W_{C_v}$ **then**
                $W_{C_v} = \alpha(u, v)$
                $R[C_v].clear()$
            end if
            if $\alpha(u, v) = W_{C_v}$ **then**
                $R[C_v].add(C_u)$
            end if
        end if
    end for
    for all $c \in C$ **do**
        if $S_c < K$ **and** $R[c].size() > 0$ **then**
            for all $x \in R[u]$ **do**
                $C.merge(c, x)$
            end for
            $flag \leftarrow$ **true**
        end if
    end for
end while

4.2.1 Complexity Analysis

In Algorithm 1, computing the Dynamic Structural Similarity for each edge in the graph (Line 1 to 2) takes worst-case time complexity of $O(T \times \alpha(G) \times |E|)$ (See section 3.1.2). Algorithms 2, 3, 4 present time complexity of $O(h \times |E|)$ where h is the number of iterations needed until convergence, i.e., the maximum height of the dendogram. In complex networks, the height of the dendogram is bounded by $h = O(log|V|)$ [15]. Thus, the time complexity of WMW is $O(T \times \alpha(G) \times |E| + \log(|V|) \times |E|)$. However, real complex networks are sparse, i.e., $|E| = O(|V|)$, and exhibit community structure. In these networks, the number of iterations T required for the DSS is small ($T \approx 5$), and also the first hierarchy of communities is found at low levels of the dendogram. For that reason, we claim that WMW presents an average-case time complexity of $O(\alpha(G) \times |E|)$ in most practical scenarios in complex networks. The space complexity is $O(|V| + |E|)$.

On the other hand, the original authors of the Weak and Most Weak community definitions have proposed algorithms to detect Weak and Most Weak communities with average time complexity of $O(|V|^2)$ in complex networks, making them impractical in large-scale scenarios [57, 34].

4.3 Overlapping Community Detection

In the seminal work developed in [11], it is shown through empirical evaluation, that the overlapping community structure in a network only differs from the disjoint part in the nodes with multiple memberships, i.e., if the overlapping nodes are removed or assigned to a single community, then the filtered community structure corresponds to that detected with classical disjoint community detection algorithms.

Following the previous idea, we propose the Overlapping Weak Most Weak Community Detection algorithm (OWMW) to detect overlapping community structure from core disjoint communities. The OWMW algorithm works with the following two-step approach:

- **Step 1**. A disjoint community structure $C(G)$ is detected with the WMW algorithm.
- **Step 2**. A fuzzy community structure $C^f(G)$ is generated by computing a membership function $f_i : C_i \mapsto [0, 1]$. Optionally, a crisp community structure $C^\alpha(G)$ can be generated from the fuzzy one by applying an α-cut to $C^f(G)$.

4.3.1 Membership Function

In order to define the membership function f_i, first we define the connectivity of a node u towards a community C.

DEFINITION 15. The connectivity of a node u towards a community C, denoted by $Conn(u, C)$, quantifies the density of connections going from the node u towards nodes in the community C as follows,

$$Conn(u, C) = \sum_{v \in C} DSS(u, v) \tag{4.1}$$

Instead of using other similarity measures to determine the connectivity of a node towards a particular community, like the *Permanence* function proposed in [11], we employ the Dynamical Structural Similarity, since it describes with high quality the density of the neighborhood that surrounds two connected nodes (That is the motivation of the *Permanence* function). This re-utilization saves time of computation, due to the DSS is computed previously in Step 1 of the OWMW algorithm. Based on the definition of connectivity, the membership function is defined as follows,

DEFINITION 16. The *Membership* of node u in community C_i, denoted by $f_i(u)$, is defined as the connectivity $Conn(u, C_i)$ normalized by the maximum connectivity in the neighborhood of u, weighted by the fraction of nodes in the community C_i that are connected to the node u, that means,

$$f_i(u) = \frac{Conn(u, C_i)}{\max\limits_{C_j \in C(G)} Conn(u, C_j)} \times \frac{|\{v \in C_i : (u, v) \in E\}|}{|C_i|} \tag{4.2}$$

In order to know how representative is $Conn(u, C_i)$, it is normalized respect to the maximum connectivity in the neighborhood of u. However, the normalized connectivity can be too high even if the number of connections towards C is low (Due to high density of connections among low number of nodes). For that reason, the normalized connectivity is weighted by the fraction of nodes in C_i that are connected to u (This weighting scheme also gives importance to the number of connections from u towards C_i). From Equation 4.2 is easy to see that $f_i(u) \in [0, 1]$.

4.3.2 ϵ-Core Community Definition

A possibility to increase the probability of identifying overlapping communities, is to detect early in-between communities in the disjoint community structure, for that reason, we propose the following ϵ-*Core Community Definition*.

DEFINITION 17. An ϵ-*Core Community* is a candidate community C in Steps 2 and 3 of the WMW algorithm, whose adjacent communities are all connected to C with similarity measure no less than $|maxS - \epsilon|$, where $maxS$ is the maximal similarity measure connecting C to some of its adjacent communities.

The WMW algorithm must be modified in Step 1 of the OWMW algorithm in order to detect both, the target community definition passed by parameter (CD) and also the

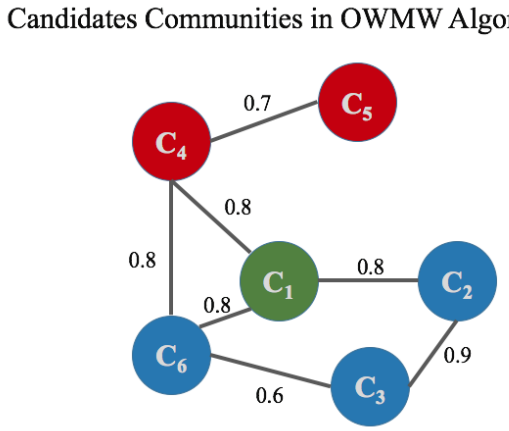


FIGURE 4.2. Six candidate communities in one iteration of the OWMW algorithm. The community in green color (C_1) is an example of an ϵ-Core Community for $\epsilon = 0$. Only the edges with maximum similarity between two communities are displayed.

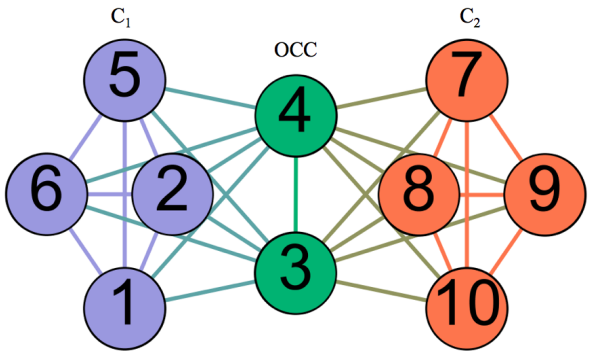


FIGURE 4.3. Toy network with two overlapping communities C_1 and C_2. The overlapping nodes between C_1 and C_2 are those in the ϵ-Core Community OCC composed of the nodes 3 and 4. The algorithm OWMW correctly assigns the nodes 3 and 4 in both communities C_1 and C_2 with maximal membership.

ϵ-Core communities. Figure 4.2 shows a visual description of an ϵ-Core Community. Additionally, Figure 4.3 shows an example of an ϵ-Core conformed by two nodes in a toy network with overlapping community structure. Thanks to the community OCC, the OWMW algorithm correctly classifies the nodes 3 and 4 as the overlapping nodes between the two disjoint communities C_1 and C_2, and assigns them in both communities with maximal membership. Also, by definition, the community OCC is neither a Weak nor a Most Weak community.

4.3.3 Complexity Analysis

The time complexity in Step 1 is dominated by the complexity of the WMW algorithm, thus Step 1 presents average time complexity of $O(\alpha(G) \times |E|)$. The time complexity in

Step 2 is dominated by the time required to compute the membership function f_i (Equation 4.2) for each node in the graph. Moreover, to compute the connectivity of a particular node u towards its adjacent communities with Equation 4.1, takes time complexity of $O(d[u])$, since querying the DSS takes $O(1)$ (because the DSS was computed in Step 1). Also, the maximum connectivity in the neighborhood of u and the fractions of nodes connecting u towards a particular community, can be tracked while computing its connectivity. So, to compute the membership function for each node in the graph takes time complexity of $\sum_{u \in V} O(d[u]) = O(|E|)$. Therefore, the OWMW algorithm presents average time complexity of $O(\alpha(G) \times |E|)$, like the original WMW algorithm. The space complexity is still $O(|V| + |E|)$.

Experiments

The experimental evaluation and validation of the WMW algorithm is performed in Section 5.3.

4.4 Summary

In this chapter, a novel algorithm for community detection in complex networks was presented. This algorithm, named the WMW algorithm, is inspired on an agglomerative hierarchical clustering technique that merges clusters based on whether they meet a community definition given as parameter, and whether they achieve a minimum community size, also given as parameter. A reference synchronous implementation of the algorithm was provided, and such implementation presents super-linear time complexity with respect to the size of the input, so it can be applied efficiently to large-scale networks. Moreover, the WMW algorithm is expected to detect hierarchical community structure because it is based on a hierarchical clustering technique, and also is expected to be resolution limit free, since it uses per-cluster quality functions, instead of partition quality functions.

Additionally, the OWMW algorithm was presented as an extension of the WMW algorithm capable of detecting overlapping community structure. The OWMW algorithm leverages the disjoint community structure generated by the original WMW algorithm to compute a fuzzy partition. The fuzzy partition is created by computing a membership function (computed from the Dynamic Structural Similarity values) for each node in the graph. Moreover, a crisp partition can be obtained from the fuzzy one by applying an α-cut. The OWMW algorithm presents the same computational complexity of the WMW algorithm, so it can be applied efficiently to large-scale networks.

CHAPTER 5

Experiments and Results

In this chapter, an extensive experimentation is performed in order to test the properties, efficiency and efficacy of the proposed algorithms and to compare them to the state-of-the-art.

This chapter is organized as follows: Section 5.1 describes the experimental settings used (algorithms, data sets, and metrics). Section 5.2 presents and discusses the experimental results obtained by the Dynamic Structural Similarity and the ISCAN algorithm. Section 5.3 presents and discusses the experimental results obtained by the WMW algorithm and the OWMW algorithm.

5.1 Experimental Settings

The experimental settings were chosen from the literature. Those settings are widely used to perform validation, evaluation and comparative analyses among different algorithms, so they provide a solid experimental framework. Additionally, several experimental settings were proposed in order to model more realistic scenarios.

5.1.1 State-of-the-art algorithms

In order to compare our proposed algorithms, a set of algorithms were chosen from the literature. Those algorithms are top because of their efficiency and efficacy in the community detection task. Only unsupervised algorithms with average time complexity of $O(|E|)$ in complex networks are considered, because they not require the number of communities to detect as parameter, and are applicable to large-scale networks. Table 5.1 summarizes the selected algorithms.

TABLE 5.1. Set of fast and unsupervised state-of-the-art algorithms selected for the comparative analysis.

Algorithm	Detect Overlapping Communities	Reference
Label Propagation (LP)	No	[58]
COPRA	Yes	[31]
SLPA	Yes	[71]
MCL	No	[22]
ATTRACTOR	No	[64]
LOUVAIN	No	[6]
SCAN	No	[72]
pSCAN	No	[12]
INFOMAP	Yes	[63]
GCE	Yes	[46]
DEMON	Yes	[20]

5.1.2 Benchmark Graphs

Several benchmark graphs widely used in the literature were selected in order to validate and evaluate our proposed algorithms, and also to compare them to the state-of-the-art. The benchmark graphs were divided into two set: real-world graphs and synthetic graphs.

5.1.2.1 Real-world Networks Dataset

A set of real-world graphs was selected from the Standford Large Network Dataset (SNAP) [47] and the Network Data Repository [62]. The set is comprised of small-to-large sized networks with and without planted ground-truth community structure. Table 5.2 shows the set of undirected unweighted real-world networks.

TABLE 5.2. Set of Small-to-large Undirected Unweighted Real-world Networks.

Network	No. Nodes	No. Edges
Zakary Karate Club [62]	34	48
Books about US Politics [62]	105	441
Ego-Facebook [47]	4039	88234
Gowalla [47]	196591	950327
Amazon [47]	334863	925872
DBLP [47]	317080	1049866
Roads PA [47]	1088092	1541898
Youtube [47]	1134890	2987624
Livejournal [62]	3997962	34681189
Facebook UCI-UNI [62]	58790782	92208195
Facebook KONECT [62]	59216211	92522012

5.1.2.2 Synthetic Networks Dataset

The LFR benchmark [45] was used to generate the set of synthetic graphs. The LFR benchmark generates unweighted and undirected graphs with planted ground-truth community structure. Also, it produces networks with node degree and community size that follow power-law distributions, making it more appropriate than the Girvan-Newman benchmark to model complex networks. By varying the mixing parameter μ, LFR can generate networks with community structure more or less difficult to identify.

Tables 5.3, 5.4 and 5.5 show the parameters used in the LFR Benchmark to generate the sets of synthetic networks. The parameter settings for each dataset are extracted from [73, 38, 70]. However, the community size distribution exponent τ_2 has been modified in the datasets LRF1 and LFR2 from -1 to -2.5 and -2 respectively, in order to model more realistic scenarios and effectively test the resolution limit of the compared algorithms.

TABLE 5.3. LFR1 - Dataset with networks of five different sizes.

Parameter	Values
Number of nodes N	233, 482, 1000, 3583, 8916
Maximum node degree $maxk$	$0.1N$
Average node degree $avgk$	20
Degree distribution τ_1	-2
Maximum community size $maxc$	$0.1N$
Minimum community size $minc$	Default
Community size distribution τ_2	-2.5
Mixing parameter μ	[0.05, 0.075] with step of 0.05
Overlapping Nodes O_n	0
Overlapping Memberships O_m	0

TABLE 5.4. LFR2 - Dataset with networks of size 1000 and 5000 with Big (B) and Small (S) communities.

Parameter	Values
Number of nodes N	1000, 5000
Maximum node degree $maxk$	50
Average node degree $avgk$	10
Degree distribution τ_1	-2
Maximum community size $maxc$	50, 100
Minimum community size $minc$	10, 20
Community size distribution τ_2	-2
Mixing parameter μ	[0.05, 0.075] with step of 0.05
Overlapping Nodes O_n	0
Overlapping Memberships O_m	0

TABLE 5.5. LFR3 - Dataset with networks of size 1000 and 5000 with Big (B) and Small (S) overlapping communities.

Parameter	Values
Number of nodes N	1000, 5000
Maximum node degree $maxk$	50
Average node degree $avgk$	10
Degree distribution τ_1	-2
Maximum community size $maxc$	50, 100
Minimum community size $minc$	10, 20
Community size distribution τ_2	-1
Mixing parameter μ	[0.05, 0.075] with step of 0.05
Overlapping Nodes O_n	500
Overlapping Memberships O_m	[1, 8] with step of 1

5.1.3 Validation and Evaluation Criteria

Many algorithms to perform community detection have been proposed in the literature [27]. For that reason, a standard and reliable framework is required to validate and evaluate the results obtained with those algorithms and to perform comparative analysis among them.

If networks with ground-truth disjoint communities[1] are provided, a community detection algorithm is validated through the sqrt-Normalized Mutual Information (NMI) [73],

$$NMI(C_1(G), C_2(G)) = \frac{\sum_{i=1}^{|C_1(G)|} \sum_{j=1}^{|C_2(G)|} N_{ij} \log(N_{ij}N/(N_{ic}N_{rj}))}{\sqrt{\sum_{i=1}^{|C_1(G)|} N_{ic} \log(N_{ic}/N) \sum_{j=1}^{|C_2(G)|} N_{rj} \log(N_{rj}/N)}} \tag{5.1}$$

The NMI compares two partitions generated from the same dataset by assigning a score within the range [0, 1], where 0 indicates that the two partitions are independent from each other and 1 if they are equal.

If the ground-truth communities contain overlapping communities, the NMI extended to covers [50] is used. If the number of communities in the covers being compared are too different, the NMI is not a good quality measure. For that reason, the Adjusted Omega Index [18] is also employed. The Adjusted Omega Index assigns a score within the range [0, 1], where 0 indicates that the two partitions are independent from each other and 1 if they are equal.

[1]The ground-truth is a planted community structure that must be detected by any algorithm on the network.

If networks without ground-truth disjoint communities are provided, a community detection algorithm is evaluated through the Modularity Score (Q) [53],

$$Q(C) = \sum_{i} \left(e_{ii} - a_i^2\right) \tag{5.2}$$

The Modularity score is equal to $Q = 0$ if the partitions are not better than a random assignment, and is equal to $Q = 1$ if the partitions present strong community structure. In complex networks, good rates of Modularity falls in the interval $[0.3, 0.7]$, higher values to 0.7 are rare and are biased towards the resolution limit [26].

5.2 Dynamic Structural Similarity

In this section several experiments are performed on real-world and synthetic benchmark graphs to show the dynamic behavior of the proposed Dynamic Structural Similarity, and how it can be used to improve the overall performance of an algorithm that performs community detection on graphs.

For all the experiments executed in this section, the DSS is initialized with Equation 3.2 using an initial similarity score $s = 1$. In order to facilitate the comparison with the LSS, we normalize $DSS(u, v)$ to the interval $[0, 1]$ by applying min-max normalization after each iteration of the Equation 3.3. In section 5.2.1, two real-wold networks extracted from [47] and a toy network are used to analyze the dynamic properties and evolution of the DSS through the time. In section 5.2.2, a set of synthetic benchmark graphs with planted ground-truth community structure is used to validate, evaluate and compare the proposed ISCAN algorithm.

5.2.1 Evolution on Real-world Complex Networks

Figure 5.1 shows the structural similarity computed in a toy network with both, the DSS and the LSS. As we can see, with one iteration of the DSS (Figure 5.1a), there are not important differences between both similarities (5.1b). However, the DSS deviates significantly from the LSS as the number of iterations increases, as shown in Table 5.6. In fact, the more number of iterations, the better characterization (in terms of the similarities) is obtained for both intra-cluster edges (e.g. 1-2, 4-6, 8-9, etc.) and inter-cluster edges (e.g. 1-10, 3-4, 6-7). Also, later iterations to iteration 14 not affect any similarity in the network, that means at iteration 14 the DSS achieves a non-trivial steady-state[2] in this sample network.

Because the toy network may not be informative enough about the dynamic properties, the DSS was applied in a real-world network. Figure 5.2 shows the dynamic behavior of

[2]A trivial steady-state is whose nodes are completely similar (maximal similarity for each edge) or completely dissimilar (minimal similarity for each edge).

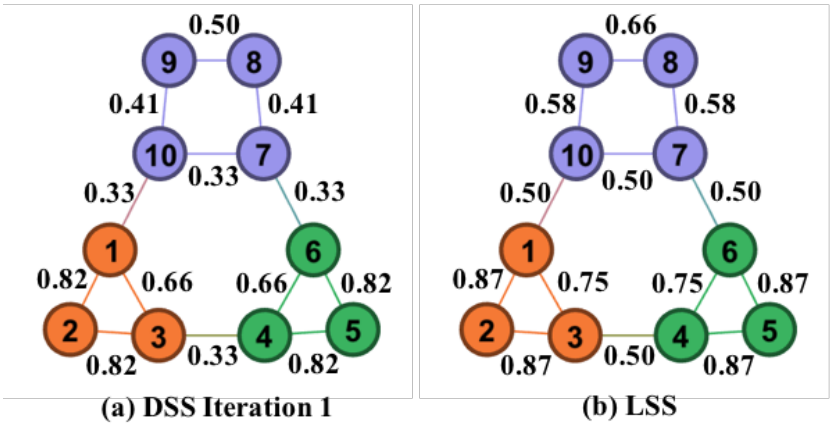


FIGURE 5.1. First iteration of the Dynamic Structural Similarity (a) and the local structural similarity (b) for each edge in a toy network.

TABLE 5.6. Evolution of the Dynamic Structural Similarity in a toy network.

Edge	Iteration					
	1	2	3	4	5	≥ 14
1-2	0.82	0.91	0.94	0.95	0.97	1.00
1-3	0.66	0.82	0.89	0.93	0.95	1.00
1-10	0.33	0.24	0.17	0.13	0.09	0.00
2-3	0.82	0.91	0.95	0.97	0.99	1.00
3-4	0.33	0.18	0.10	0.05	0.02	0.00
4-5	0.82	0.91	0.95	0.97	0.99	1.00
4-6	0.66	0.82	0.89	0.93	0.95	1.00
5-6	0.82	0.91	0.94	0.95	0.97	1.00
6-7	0.33	0.24	0.17	0.13	0.09	0.00
7-10	0.33	0.31	0.32	0.35	0.38	0.50
7-8	0.41	0.41	0.43	0.45	0.46	0.50
8-9	0.50	0.55	0.57	0.57	0.56	0.50
9-10	0.41	0.41	0.43	0.45	0.46	0.50

the DSS in an ego-network. The ego-network consists of friends lists from Facebook. This network was collected from survey participants using a Facebook app and is conformed by 4039 nodes and 88234 edges [47]. As we can see, the proposed structural similarity behave as a dynamical system in the ego-network, presenting many fluctuations (i.e., edges whose similarity decrease/increase in the iteration t and increase/decrease on iteration $t+1$) at early iterations and reaching a non-trivial steady-state on later iterations[3]. This behavior differs from other dynamical similarities in the literature, consider for example the SimRank [36], where the similarities increase monotonically after each iteration (no fluctuation occurs and no similarity decreases). We consider important a dynamic behavior because if the tendency in the similarity changes is known in advance, then the iterative

[3]For the experiment with the ego-network, we consider stable any variation in the similarity below 1e-12. This threshold is no required to achieve the convergence, but it allows us to summarize the dynamic process with fewer iterations.

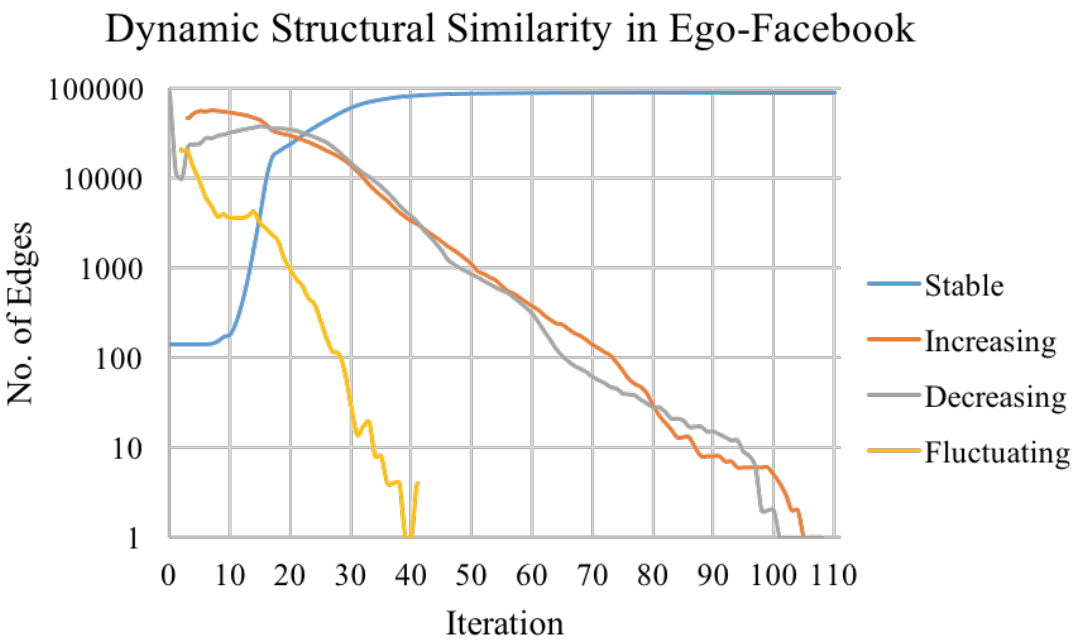


FIGURE 5.2. Number of stable/increasing/decreasing/fluctuating edges (in log-scale) in the ego-facebook network [47] in function of the iteration of Equation 3.3. This plot shows the dynamic behavior of the proposed structural similarity in a real-world network. The majority of fluctuations occurs at early iterations and the system converge to a non-trivial steady-state.

process becomes less informative. Another interesting result from this experiment is the capability of our proposal to converge to a non-trivial steady-state without requiring a cooling factor (e.g. the constant C in SimRank or the damping factor α in the Personalized Page Rank), this property allows the DSS to be parameter free.

In order to see with more detail the changes in the DSS through the time, a sample of fourteen edges was taken from the Youtube network 5.2. The Youtube network is conformed by 1134890 nodes and 2987624 edges, each node represents an users and each edge represents an user-to-user friendship.

Figure 5.3 shows the changes in the DSS for each sample edge as the number of iterations increases. Several conclusions about the behavior of the DSS can be drawn: First, the initial values of the DSS can be loosely related to that values in later iterations, i.e., edges with high initial DSS can abruptly decrease (e.g. Figure 5.3b) and edges with low initial DSS can suddenly increase (e.g. Figure 5.3c). Second, The edges can fluctuate their DSS as the system evolves (e.g. Figure 5.3a and edges 88903-391756, 4168240-581532 in Figure 5.3b), with important fluctuations being observed at early iterations ($T \leq 16$). Third, the changes in the DSS occur at different rates, with some edges increasing/decreasing faster than others. Fourth, different edges with equal initial DSS not necessarily behave in the same manner as the system evolves (e.g. edges 416824-581532,

860514-981122 in Figure 5.3b and edges 201913-491692, 1067384-1067420 in Figure 5.3c). Fifth, all the sample edges tend to converge to a non-trivial steady-state.

5.2.2 ISCAN - Improved SCAN Algorithm

For this experiment, 30 LFR networks were generated for each combination of parameters listed in Table 5.4. Also, for both SCAN and ISCAN, the parameter minimum number of points was set to $\mu = 0.2$. Additionally, the number of iterations for the DSS in ISCAN was set to $T = 5$.

5.2.2.1 Sensitivity to Parameter ϵ

The main drawback in the usability of the SCAN algorithm is the high sensitivity of the resulting clustering to the variations in the parameter ϵ. This sensitivity was inherited from its predecessor algorithm DBSCAN[4], developed by the same author.

In order to compare the sensitivity to the parameter ϵ of SCAN and ISCAN, both algorithms were executed over the 30 LFR networks generated for each combination of parameters 5000S, 5000B and mixing parameters $\mu = \{0.35, 0.50\}$. For both SCAN and ISCAN, the parameter ϵ was varied in the interval $[0.1, 0.9]$ with steps of 0.05. The mean and standard deviation of the Modularity score were computed to measure the quality of the results. As we can see in Figure 5.4, ISCAN presents better Modularity scores compared to SCAN for any variation of the parameter ϵ. The SCAN algorithm increases effectively the Modularity from $\epsilon = 0.1$ up-to $\epsilon = 0.2$, moment in which it achieves its maximum value, but for $\epsilon > 0.2$ SCAN shows an abrupt decay in its performance, due to its high sensitivity to the parameter ϵ. In contrast, ISCAN starts with maximum values of Modularity for $\epsilon = 0.1$ and decreases continuously as ϵ increases. However, ISCAN has mitigated its sensitivity to the parameter ϵ.

5.2.2.2 Detection of Ground-truth Communities

In order to test the capacity of SCAN and ISCAN to detect ground-truth communities, both algorithms were executed on the 30 LFR networks generated for each combination of parameters listed in Table 5.4. The mean and standard deviation of the NMI were computed to measure the quality of the results. For both SCAN and ISCAN, the parameter $\epsilon = 0.2$ was fixed. The parameter ϵ was chosen based on the best average performance obtained in the benchmark graphs in Figure 5.4.

As we can see in Figure 5.5, the critical point on the performance for both algorithms arrives when the mixing parameter $\mu \geq 0.5$. Anyway, ISCAN surpass the quality of the results of SCAN in the majority of scenarios, with up-to 8% of increase in the quality of

[4]The SCAN algorithm is an adaptation to the context of graphs of the DBSCAN algorithm that originally performs density based clustering on spatial data.

the results (NMI). Also, both algorithms remain stable for fixed parameters, presenting very low standard deviation, with $STD \leq 0.05$ in all scenarios.

In order to test the sensitivity to the parameter ϵ of SCAN and ISCAN in the detection of ground-truth communities, both algorithms were executed on the 30 LFR networks generated for each combination of parameters 5000S and 5000B from Table 5.4. For both SCAN and ISCAN, the parameter ϵ was varied in the interval $[0.2, 0.5]$ with steps of 0.1. The mean of the NMI score was computed.

As we can see in Figure 5.6, the quality of the results obtained with ISCAN are less sensitive compared to SCAN for any variation of the parameter ϵ. Even though the NMI drops for both algorithms as ϵ increases, the NMI drops more rapidly in the case of SCAN.

5.2.2.3 Community Size Distribution

A recurrent problem in the community structure generated by the SCAN algorithm is the high number of singleton communities (communities with size 1) that are detected as hubs or outliers, even if they do not exhibit such characteristic. In order to compare the community size distributions generated by SCAN and ISCAN, both algorithms were executed on a sample[5] network 5000S with mixing parameter $\mu = 0.5$ taken from the 30 networks generated for each combination of parameters listed in Table 5.4. In this sample network the community size distribution was within the range $[10, 50]$. For both, SCAN and ISCAN the parameter ϵ was tested in $\{0.2, 0.35\}$. The Mean Square Error (MSE) of the estimated community size distribution respect to the real distribution was computed.

Figure 5.7a shows the community size distribution generated by SCAN and ISCAN with parameter $\epsilon = 0.35$. For this parameter setting, ISCAN obtains NMI score of 0.99 and SCAN obtains NMI score of 0.87. SCAN generates a community size distribution within the range $[1, 17]$ with more than 1400 singleton communities, resulting in a poor approximation with MSE of 42194.56, respect to the real community size distribution. On the other hand, ISCAN generates community size distribution in the range $[1, 39]$ giving a better approximation with MSE of 26.14, respect to the real community size distribution. Figure 5.7b shows the community size distribution generated by SCAN and ISCAN with parameter $\epsilon = 0.2$. In this case, ISCAN and SCAN obtain NMI score of 0.99 and 0.97 respectively. Also, SCAN reduces the number of singleton communities to 33 but they are still high compared to the 8 singleton communities generated by ISCAN. In this configuration, SCAN obtains MSE of 28.66 and ISCAN obtains MSE of 3.28. This experiment evidence one more time the high sensitivity of SCAN to small variations in the parameter ϵ, and how such sensitivity is mitigated with ISCAN. Moreover, in both cases ISCAN presents better estimation than SCAN of the ground-truth community size distribution.

[5]We take a sample network, but the experimental results follow the same trend in the remaining networks.

5.3 Community Detection Algorithm

In this section several experiments are performed on real-world and synthetic benchmark graphs. The experiments are aimed to test the properties, efficacy and efficiency of the proposed community detection algorithm on both, disjoint and overlapping community detection.

5.3.1 Detection of Ground-truth Communities

In order to test the capacity of the WMW algorithm to detect ground-truth communities, it was executed on the 30 LFR networks generated for each combination of parameters listed in Table 5.3. The mean and standard deviation of the NMI were computed to measure the quality of the results. Moreover, the parameters of WMW were set to minimum community size $K = 2$ and community definition CD =MOST WEAK. The default parameter settings were used for the other algorithms.

As we can see in Figure 5.8, the critical point on the performance for the majority of the algorithms, arrives when the mixing parameter $\mu > 0.5$. LP and COPRA present good performance until $\mu \approx 0.4$, but after that point they become particularly erratic, presenting high standard deviation. This unstable behavior is due to the random nature of the diffusion process performed by those algorithms. The structural clustering algorithms SCAN, pSCAN and ISCAN offer stable results, but the NMI starts to decrease from low values of μ. However, ISCAN effectively improve the NMI compared to SCAN and pSCAN for each test scenarios, evidencing again the advantages of using the DSS to model dense sub-graphs. Moreover, the LOUVAIN algorithm performance is affected notably when the size of the network increases, due to its resolution limit. INFOMAP is the best performer algorithm until $\mu = 0.6$, offering almost perfect and stable results, but for values of $\mu > 0.6$ it shows an abrupt decay in its performance. In this experiment MCL proof experimentally its efficacy and stability to perform community detection, since it outperforms the majority of algorithms in the test scenarios. ATTRACTOR is a top algorithm since it offers high quality and stable results, however it did not finish the test for the networks of size 8916 (Figure 5.8c) in a time gap of nine hours. This bad result is due to the convergence problems presented by its underlying dynamic interaction system, even in small networks (a size of 8916 nodes is considered small).

On the other hand, WMW performs near to the optimal with stable results while $\mu \leq 0.5$, with exception of networks with size 233 where it presents its worst rates of NMI. Moreover, WMW outperforms the other algorithms in the majority of scenarios for values $\mu > 0.6$. WMW has not been tested with the parameter CD =WEAK because for values of $\mu > 0.5$ no community in the generated ground truth meets the WEAK definition (making the ground truth undetectable for this parameter setting), in consequence WMW will presents an abrupt change of behavior after the critical point, as evidenced in [10]. In

contrast, WMW presents a smooth transition after the critical point by using the MOST WEAK definition.

5.3.2 Random Networks

In order to test the behavior of the algorithms on random networks, they were executed on networks generated with the Erdős-Rényi model and scale-free Barabási-Albert model. Networks generated using these two models are expected to have no community structure. For this experiment, the size of the networks in both models was set to 1000 nodes. For the Erdős-Rényi model the probability of connection between two nodes was varied in the interval [0.0, 0.8] with step of 0.1. For the scale-free Barabási-Albert, the parameter m was varied in the interval [0, 50] with step of 5. Moreover, 30 random networks were generated for each combination of parameters, and the mean of the number of detected communities by each algorithm was computed as measure of the quality of the results.

As we can see in Figure 5.9(a), as expected, all the algorithms detect a single big community or detect each node in the network as singleton communities. Neither a big one community nor singleton communities are considered representative community structure. However, in the case of scale-free networks (Figure 5.9b) several algorithms (Specially ISCAN and MCL) tend to detect community structure. This behavior is possible due to random fluctuations that can generate pseudo-communities in random networks.

5.3.3 Sensitivity to Parameter Settings

WMW was tested on a set of real-world networks listed in Table 5.2, by varying the parameter K in the interval $[2, 500]$ and the parameter CD in the values WEAK, MOST WEAK. Figure 5.10. shows the modularity Q computed for each combination of parameters-networks. As we can see, by increasing the parameter K, the tendency is to obtain partitions with higher values of modularity (this happens because the small communities merge into larger ones). For the Ego-network, the modularity decreases on intervals for values of k ¿ 200, possibly because the network becomes small with respect to the value of K. Moreover, all the resulting partitions present good modularity rates, since their values are above 0.6. If higher values of modularity are desired as result, then sampling higher values of K can be a good search strategy. However, with a default value of $K = 2$ it is enough to detect a first relevant hierarchy of communities. An equal tendency was obtained for CD =MOST WEAK.

Figure 5.11 shows the number of detected communities for each combination of parameters-networks. For each case, the number of communities decreases following a power-law distribution. This result is related to the community size distribution obtained in these complex networks, that also follows a power law distribution (See Section 5.3.4).

5.3.4 Multi-scale Community Detection

Figure 5.12 shows the community size power-law distribution obtained by WMW with parameters $K = 2$ and $CD = MOSTWEAK$ in several large-scale networks. This result indicates that WMW can effectively detect communities at any scale independent of the size of the network, from very small communities to very larges ones. This capacity lacks on algorithms that suffer of resolution limit (e.g., LOUVAIN).

5.3.5 Resolution Limit on Clique Rings

In order to check the resolution limit of the algorithms in a standard scenario, all the algorithms were executed with default parameters on two benchmark ring networks composed of N identical cliques (cliques of size 3 and cliques of size 4) connected by a single edge [26]. In those ring networks each clique is clearly considered a separate community.

Figure 5.13 shows the results obtained for each algorithm on the clique ring networks. WMW, ISCAN, SCAN, pSCAN, ATTRACTOR and MCL are able to identify all cliques as separate communities in both cases. INFOMAP presents problems on identifying the communities in the 3-Clique ring when the number of cliques in the network is greater than 21, but it obtains a perfect score in the 4-Clique ring. LP, SLPA and COPRA are problematic on both cases because they mostly underestimate the number of communities. However, COPRA overestimates the number of communities in the 4-Clique ring because it detects multiples communities in single clique structures. The bad results obtained by LOUVAIN are expected due to its resolution limit, making it the worst performer algorithm in the clique ring networks.

5.3.6 Hierarchical Community Detection

WMW is inspired in an agglomerative hierarchical technique, for that reason, it is expected to detect hierarchical community structure, i.e., bigger communities composed of smaller communities. In order to test the capacity of WMW to detect hierarchical community structure, it was executed on a hierarchical complex network that follows the Ravasz-Barabási model [60]. The network was composed of two hierarchical levels, with 25 communities (5 nodes per community) in the first level, and 5 communities in the second level. Also, WMW was tested on two small real-world networks taken from Table 5.2.

Figure 5.14 shows the two hierarchical levels identified by WMW(K=2, CD=MOST WEAK) and WMW(K=6, CD=MOST WEAK). The ground truth communities are correctly identified in both levels.

Zachary Karate Club: The data was collected from an university karate club. Each node represents a member of the club, and each edge represents a tie between two members of the club. The club was split into two groups after an argument between the administrator and the instructor [75]. Figure 5.15 shows the six communities detected

by WMW(K=2, CD=MOST WEAK) and the two communities detected by WMW(K=8, CD=MOST WEAK) that correspond to the actual split of the group after the argument.

Books About US Politics: A network of books about US politics sold by the online bookseller Amazon.com. Edges between books represent frequent co-purchasing of books by the same buyers. The books are divided into three factions: liberal, conservative and neutral [42]. Figure 5.16 shows the communities detected by WMW for three different values of K. As the value of K increases, the detected community structure better approximates to the division into the three factions previously mentioned.

From this experiment we conclude that an optimal strategy to detect the next hierarchical level in a network is by setting the parameter K to $K_{level+1} = MinCommunitySize_{level} + 1$.

5.3.7 Performance on Large-scale Complex Networks

To test WMW at scale, it was executed on several large real-world networks from Table 5.2. These networks were extracted from popular social networks such as Facebook, Youtube, Amazon, and others. To keep the comparison as fair as possible with the other algorithms, we only used the reference C/C++ implementations provided by the original authors, or those implementations provided by the C-igraph library [21]. Moreover, all the algorithms were running with default parameters in a machine with 32 GB RAM and 3.4 GHz CPU.

Table 5.7 shows the running time in seconds and Table 5.8 shows the modularity score, obtained after having executed each algorithm on the test networks. The results not obtained in a time gap of 11 hours are marked –. WMW obtains for each case good rates of modularity and low running time compared to the other algorithms. Even though MCL and INFOMAP claim to be linear time algorithms, they exhibit experimentally long running times, and are not capable to process the larger networks in the time required gap. pSCAN and ATTRACTOR algorithms are not present in the results, because the implementation provided by the authors crash when executed on the test data. However, the results obtained for pSCAN are expected to be the same as SCAN since they compute the same structural clusters, and ATTRACTOR already proved to be slow even on small networks due to its convergence problems (See 5.3.1).

This experiment let us to conclude that WMW provides the best trade-off between quality of the detected community structure and speed of computation, so it can be considered a good option to perform community detection at scale.

5.3.8 Overlapping Community Detection

In order to test the capacity of the OWMW algorithm to detect ground-truth overlapping communities, it was executed on the 30 LFR networks generated for each combination of parameters listed in Table 5.5. To get consensus on the results, the mean and standard

Table 5.7. Running Time of WMW on Large-scale Social Networks.

Algorithm	**Time (secs)**					
	Gowalla	Amazon	Youtube	Livejournal	FB UCI-UNI	FB KONECT
WMW	3.42	1.91	15.19	67.04	359	296
INFOMAP	23.36	38.09	91.60	1844.99	–	–
LP	7.73	34.90	77.77	732.10	32839.3	39117.5
MCL	1498.59	217.14	11952.73	–	–	–
LOUVAIN	7.94	8.90	17.89	241.01	1047.53	831.76
SCAN	1.01	1.76	5.82	26.43	83.72	71.72
ISCAN	3.77	2.41	12.34	65.54	183.43	146.68

Table 5.8. Modularity of WMW on Large-scale Social Networks.

Algorithm	**Modularity Q**					
	Gowalla	Amazon	Youtube	Livejournal	FB UCI-UNI	FB KONECT
WMW	0.54	0.70	0.55	0.56	0.69	0.69
INFOMAP	0.54	0.42	0.69	0.02	–	–
LP	0.50	0.78	0.66	0.48	0.63	0.63
MCL	0.28	0.67	0.33	–	–	–
LOUVAIN	0.68	0.92	0.71	0.74	0.90	0.90
SCAN	0.22	0.59	0.14	0.24	0.00	0.01
ISCAN	0.39	0.67	0.28	0.39	0.05	0.05

deviation of the O-NMI and also the Adjusted Omega Index were computed to measure the quality of the results. For this experiment, every algorithm was executed on each test network for different parameter settings, and only the best result obtained among all the parameter settings was averaged in the result. The parameter settings used for each algorithms are the following: For OWMW the parameters were fixed to $K = 2$, CD =MOST WEAK and $T = 5$. Additionally, the crisp threshold was varied for [0.005, 0.01, 0.02, 0.03, 0.04, 0.05]. For SLPA, the crisp threshold was varied for [0.05, 0.1, 0.2, 0.3, 0.4, 0.5]. For COPRA, the maximum number of membership per node was varied in the interval [1, 8]. For DEMON, the crisp threshold was varied for [0.1, 0.15, 0.2, 0.25, 0.3, 0.35]. For GCE the parameter FitnessExponent was varied for [0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4] and the other parameters were set to their default values. INFOMAP was executed with the flag *–overlapping*.

As we can see in Figure 5.17 OWMW obtains consistently the best averages of O-NMI for values of $O_m \geq 3$ in networks with mixing parameter $\mu = 0.1$. For mixing parameter $\mu = 0.3$, OWMM is outperformed by other algorithms, however OWMW is still a competitive algorithm. All the algorithms present a similar behavior as O_m increases and their performance is relatively close, with exception of INFOMAP that presents poor results in all the test scenarios.

A similar tendency can be observed in Figure 5.18. In this case OWMW presents the best average performance in terms of the Omega Index for values of $O_m \geq 3$ in all scenarios, with exception of 5.18d where it is still competitive. Moreover, in networks with

mixing parameter $\mu = 0.1$ and for values of $O_m \geq 5$, the performance of OWMW deviates significantly from the others by achieving an improvement of up-to 20 percent in the best case.

In all the test scenarios, the majority of algorithms provide stable results, with standard deviation below 0.03.

5.4 Summary

In this chapter, an extensive experimentation was executed in order to test the properties, efficiency and efficacy of the proposed algorithms, and to compare them with the state-of-the-art. The experimental settings are widely used in the literature to perform validation, evaluation and comparative analysis of community detection algorithms, so they provide a solid experimental framework.

Based on the empirical and statistical evidence provided by the experimentation, several assumptions and expectations about the proposed algorithms could be confirmed. For example, it was confirmed that the Dynamic Structural Similarity is in fact a dynamic system. Also, it was verified that thanks to the Dynamic Structural Similarity, ISCAN effectively outperforms the quality of the community structure provided by SCAN, and also reduces its sensitivity to the parameter settings. Moreover, it was shown that WMW can detect hierarchical and multi-scale community structure, and can be applied efficiently in large-scale scenarios. Also, it was shown that both algorithms WMW and OWMW, are competitive with the state-of-the-art in terms of quality of the detected community structure, and they present superior performance several test scenarios.

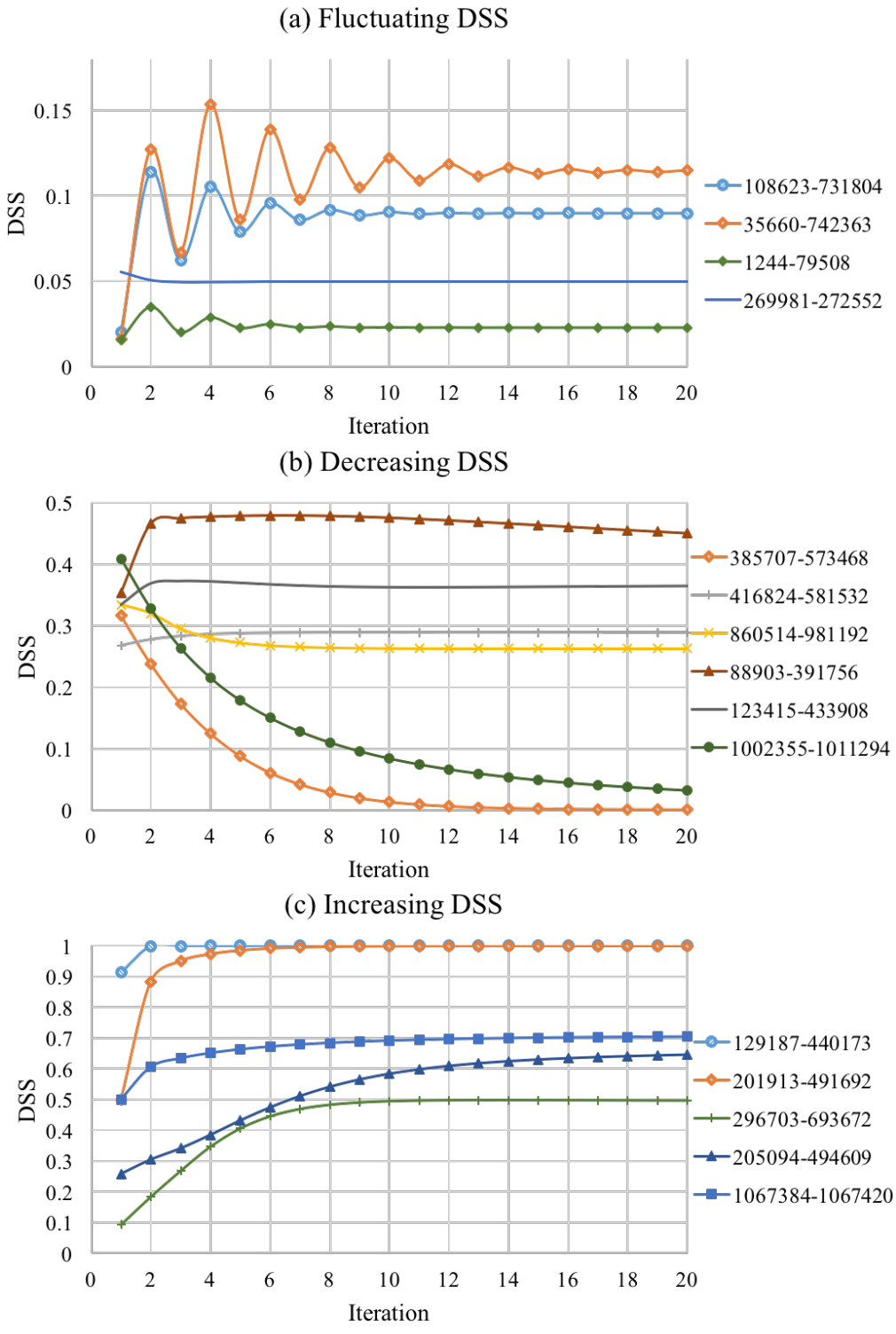


FIGURE 5.3. Dynamic Structural Similarity in function of the iteration for a sample of fourteen edges taken from the Youtube network [47]. Each series corresponds to an edge in the network.

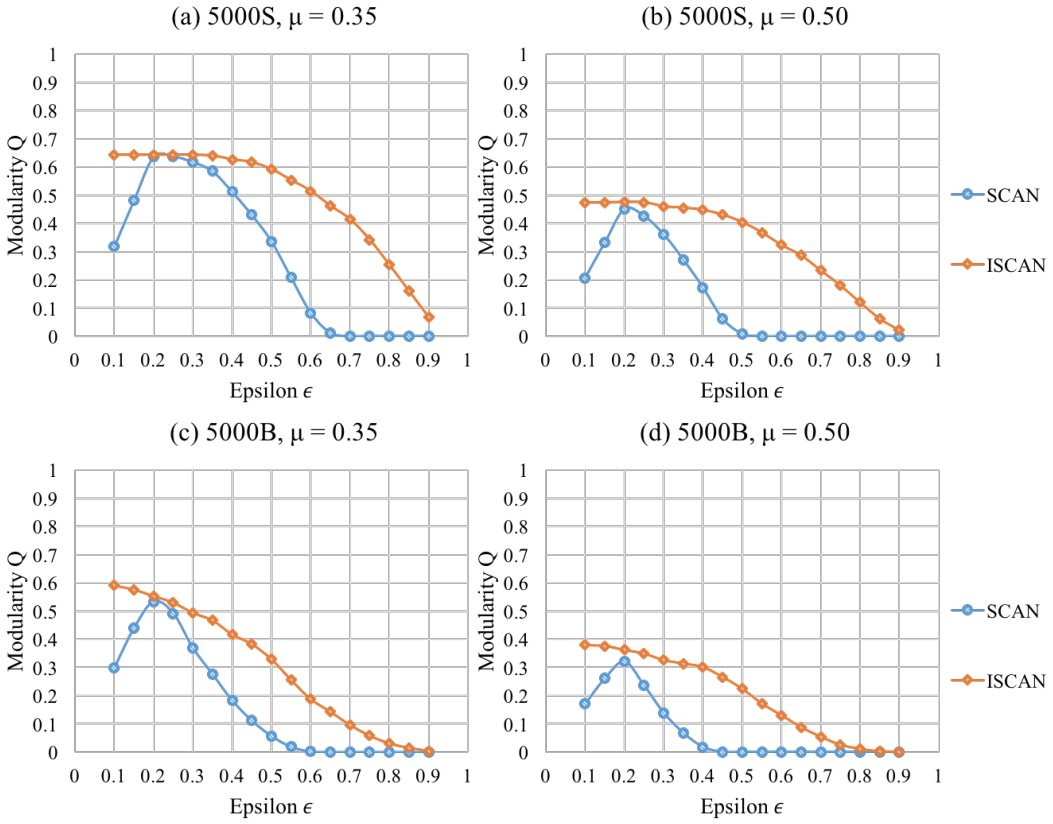


FIGURE 5.4. Mean value of the Modularity score (higher values are better) in function of the parameter epsilon ϵ for the algorithms SCAN and ISCAN. The algorithms were executed on networks 5000S and 5000B with mixing parameters $\mu = \{0.35, 0.50\}$ (higher mixing parameter creates harder decision tasks).

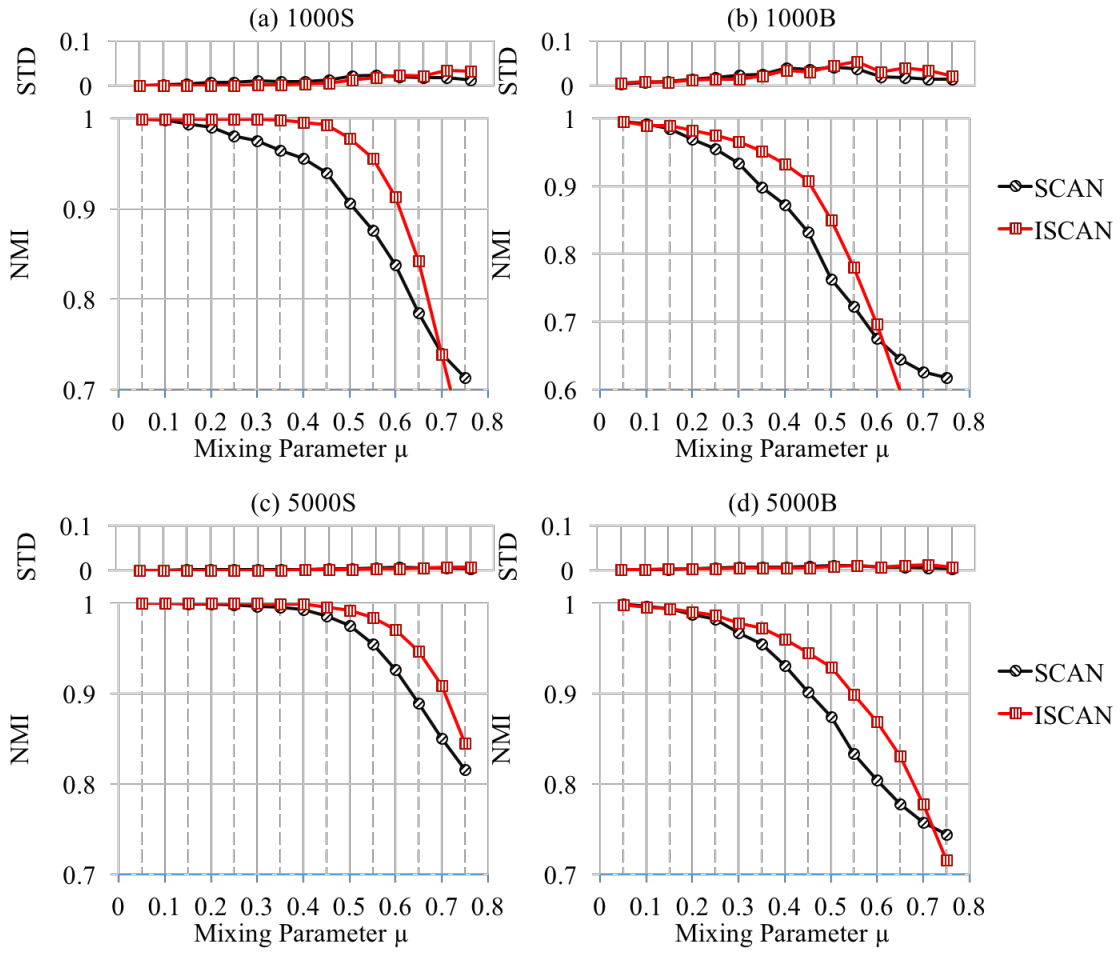


FIGURE 5.5. (Lower row) Mean value of the sqrt-Normalized Mutual Information NMI (higher values are better) as function of the mixing parameter μ. (Upper row) The standard deviation STD (lower values are better) of the NMI as function of μ.

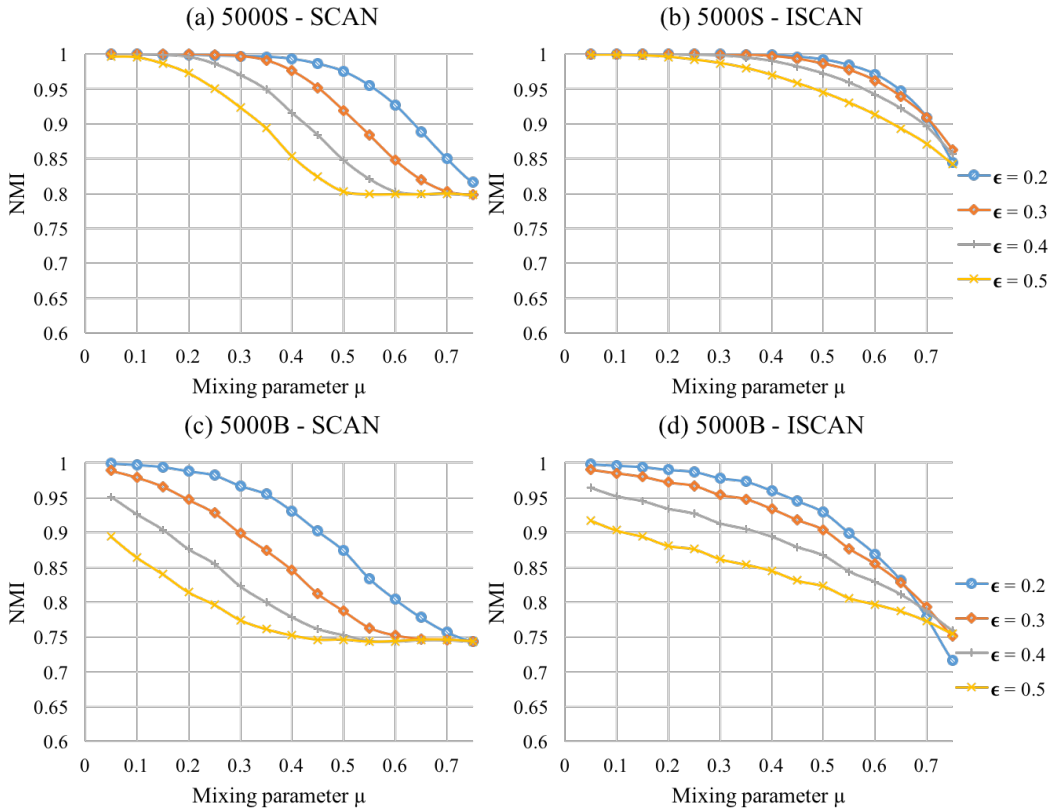


FIGURE 5.6. Mean value of the sqrt-Normalized Mutual Information NMI (higher values are better) as function of the mixing parameter μ. Series are plotted for different values of the parameter ϵ in the interval $[0.2, 0.5]$.

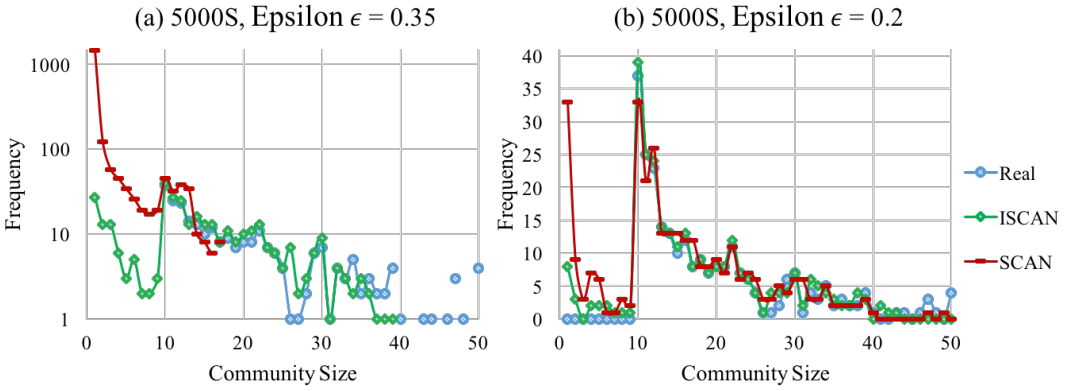


FIGURE 5.7. The real community size distribution and the estimations given by the algorithms on a sample network 5000S with mixing parameter $\mu = 0.5$. The real community size distribution contains community sizes within the range [10, 50].

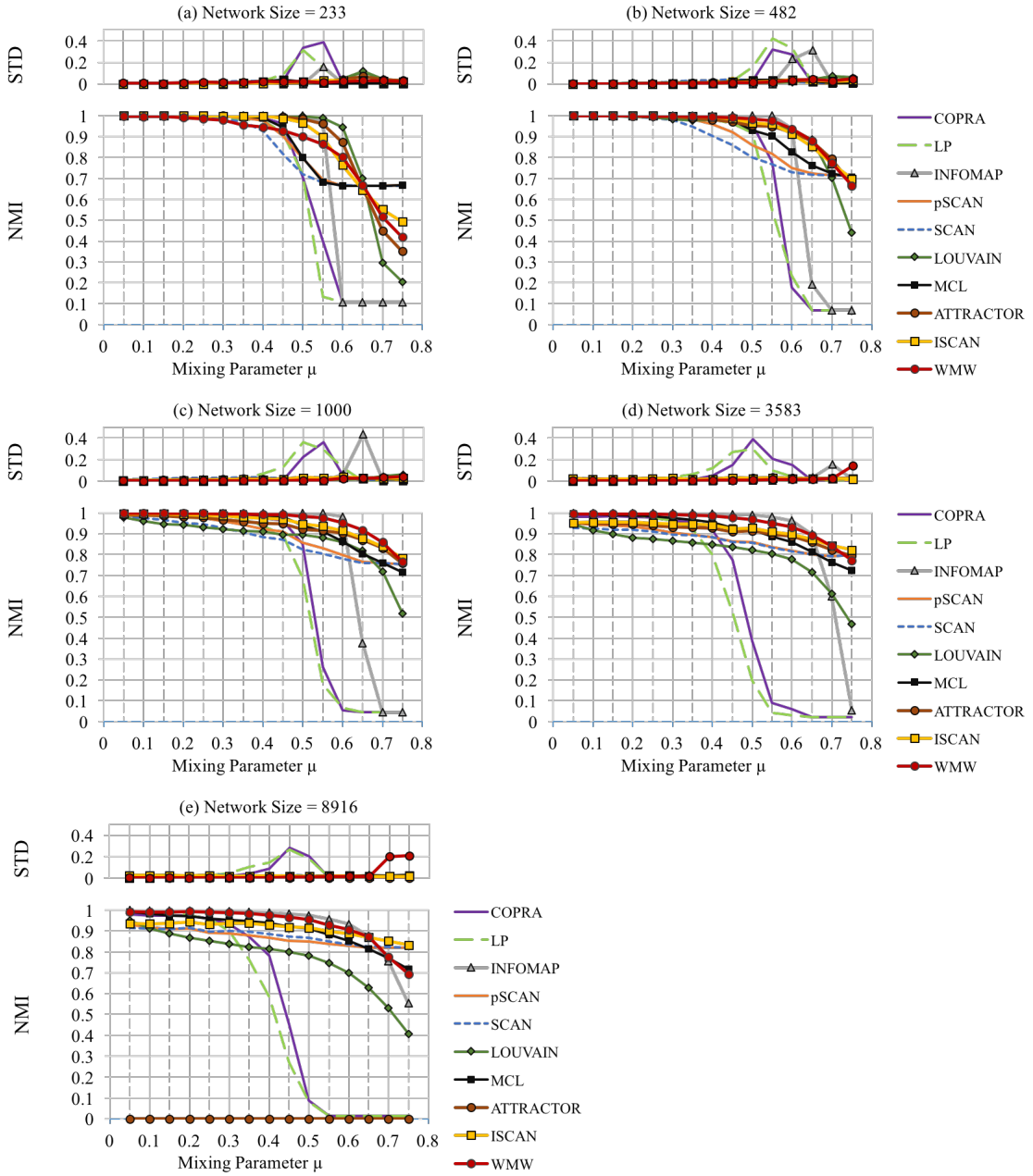


FIGURE 5.8. (Lower row) Mean value of the sqrt-Normalized Mutual Information NMI (higher values are better) as function of the mixing parameter μ. (Upper row) The standard deviation STD (lower values are better) of the NMI as function of μ.

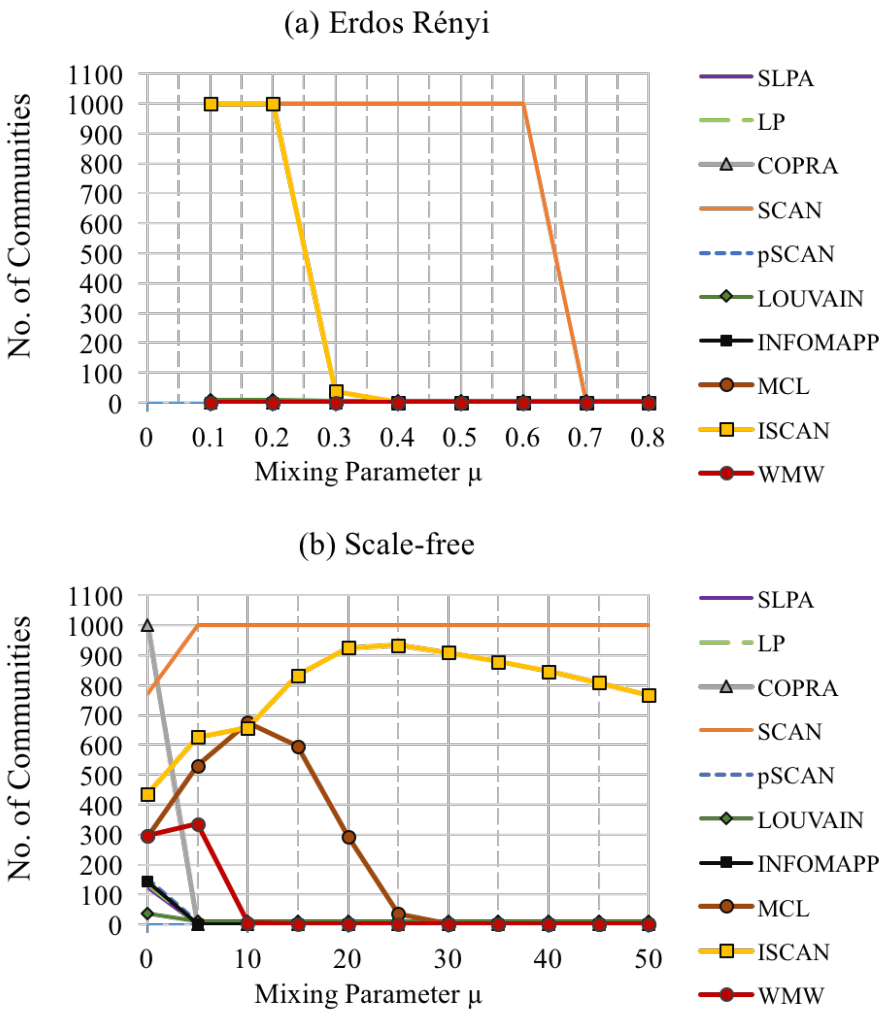


FIGURE 5.9. Mean number of detected communities in random networks (values closer to zero or 1000 are better). These networks are generated with the random models Erdős-Rényi and Scale-free Barabási-Albert.

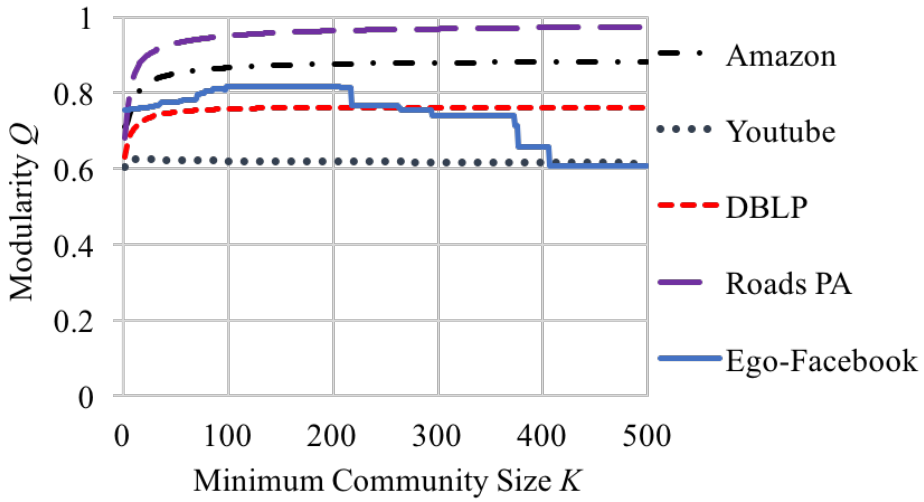


FIGURE 5.10. Modularity rate Q (Higher values are better) in function of $WMW(K, CD = WEAK)$ for different values of K.

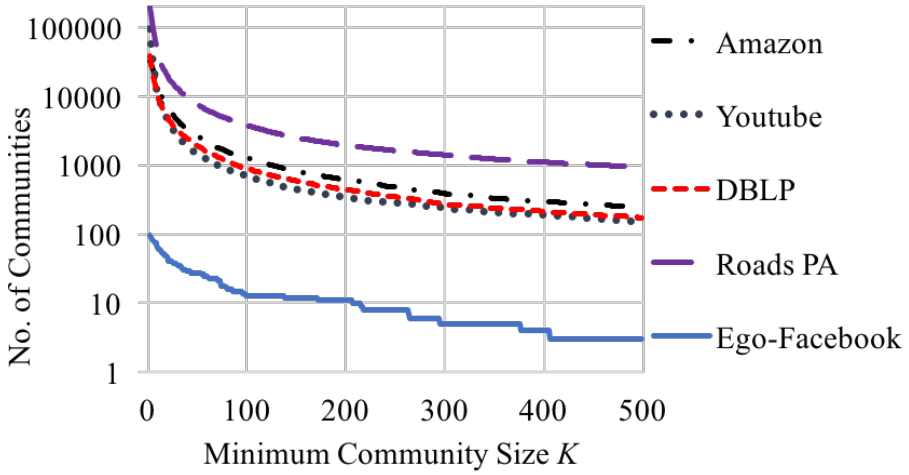


FIGURE 5.11. Number of detected communities (in log-scale) in function of $WMW(K, CD = WEAK)$ for different values of K.

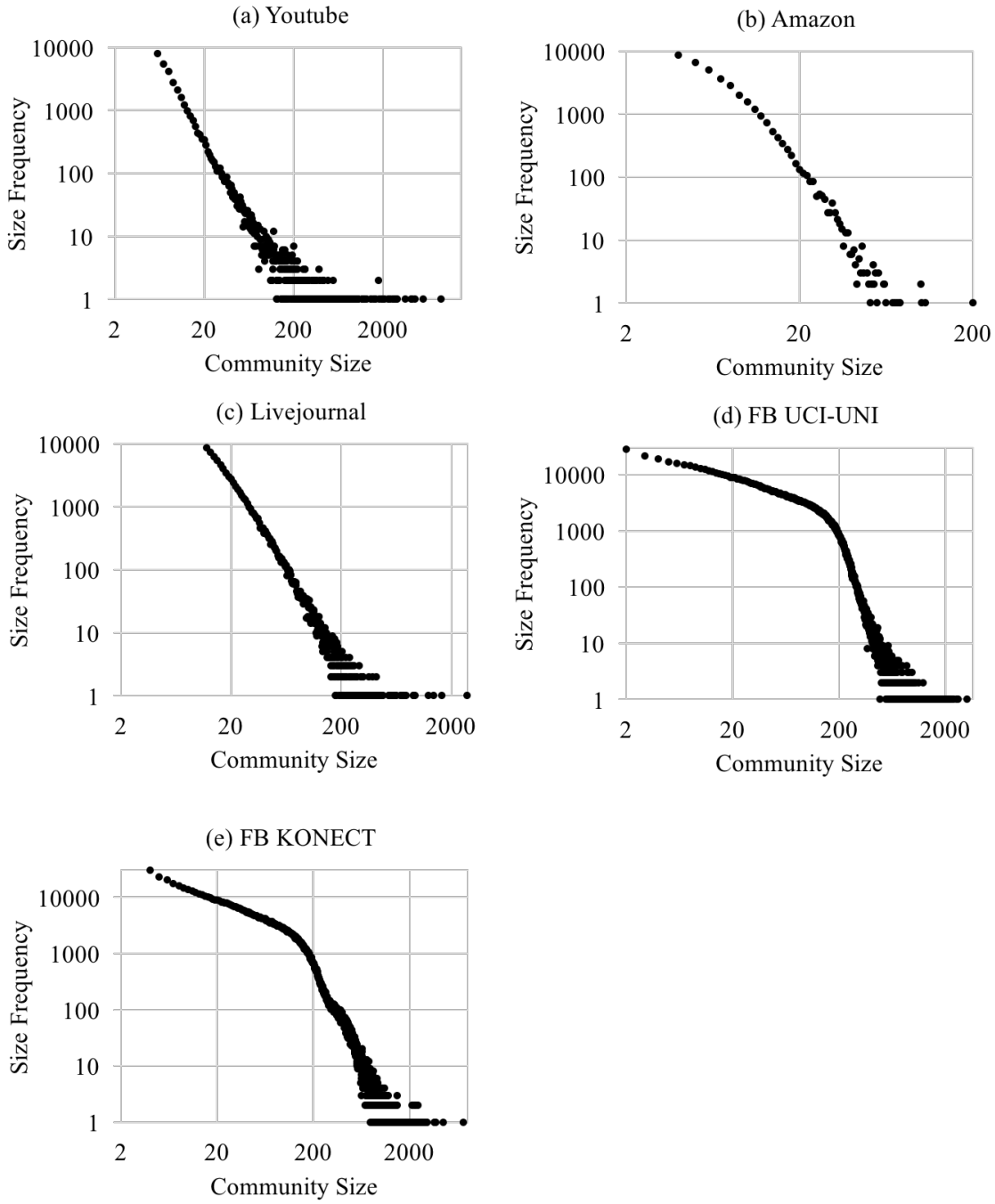


FIGURE 5.12. Community size distribution obtained with $WMW(K = 2, CD = MOSTWEAK)$ in several large-scale real-world networks (See Section 5.1). The results fit to power-law distributions with exponents: (a) -1.30, (b) -3.1, (c) -2.4, (d) -2.6 (e) -2.5.

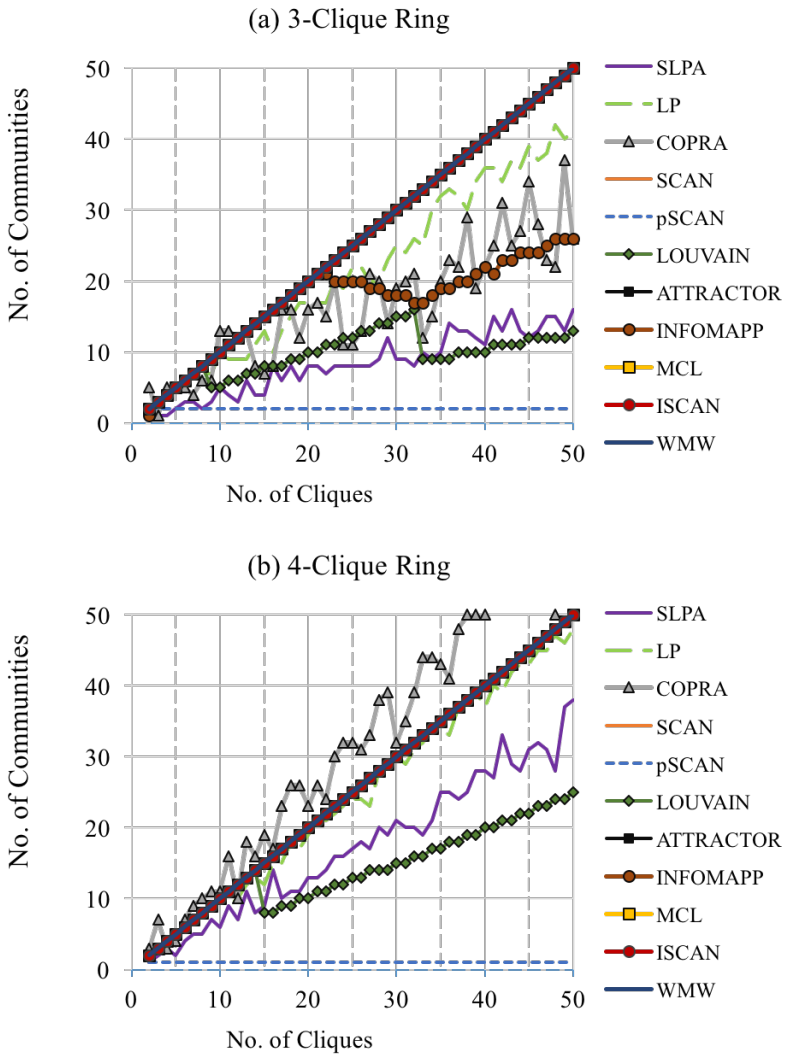


FIGURE 5.13. Test of resolution limit. Number of detected communities (values closer to the number of cliques are better) on the ring networks composed of identical cliques connected by a single edge.

(a) $K = 2$ (b) $K = 6$

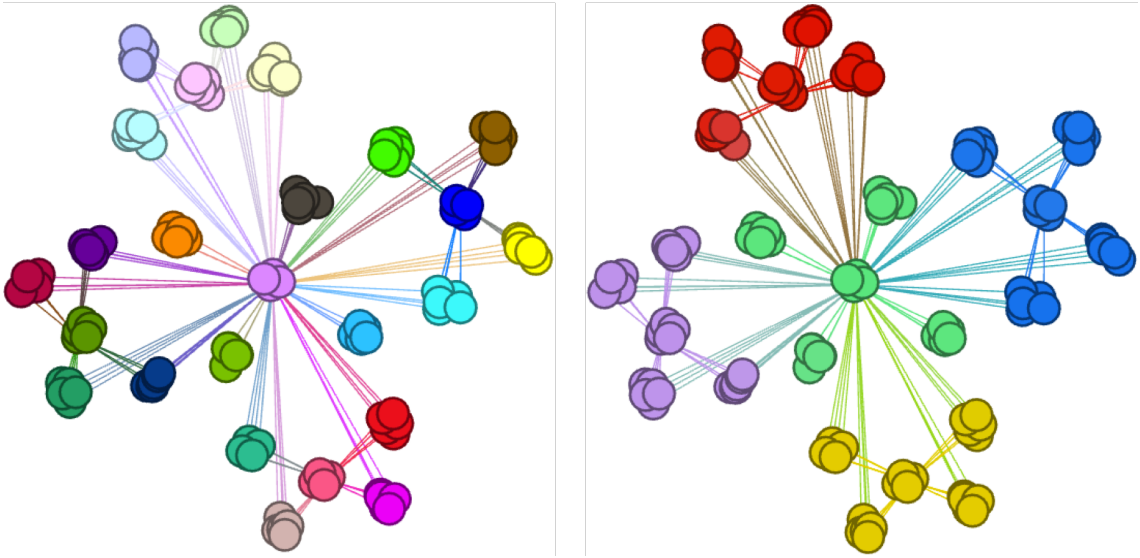


FIGURE 5.14. Communities detected by WMW in a Ravasz-Barabási complex network by tuning the parameter minimum community size K. The network is composed of two hierarchical levels correctly identified by the algorithm.

(a) $K = 2$ (b) $K = 8$

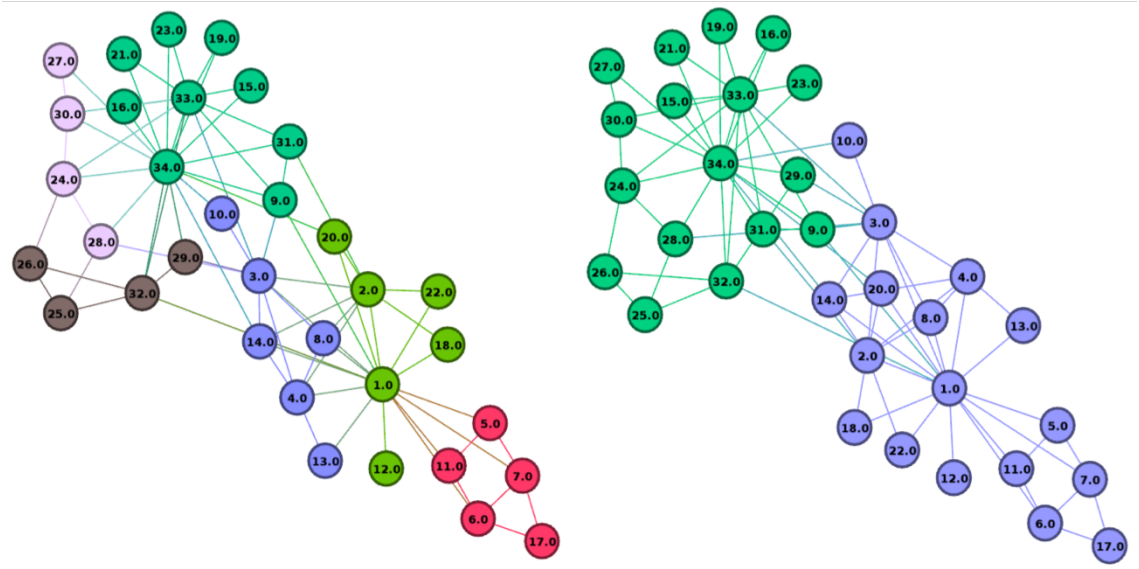
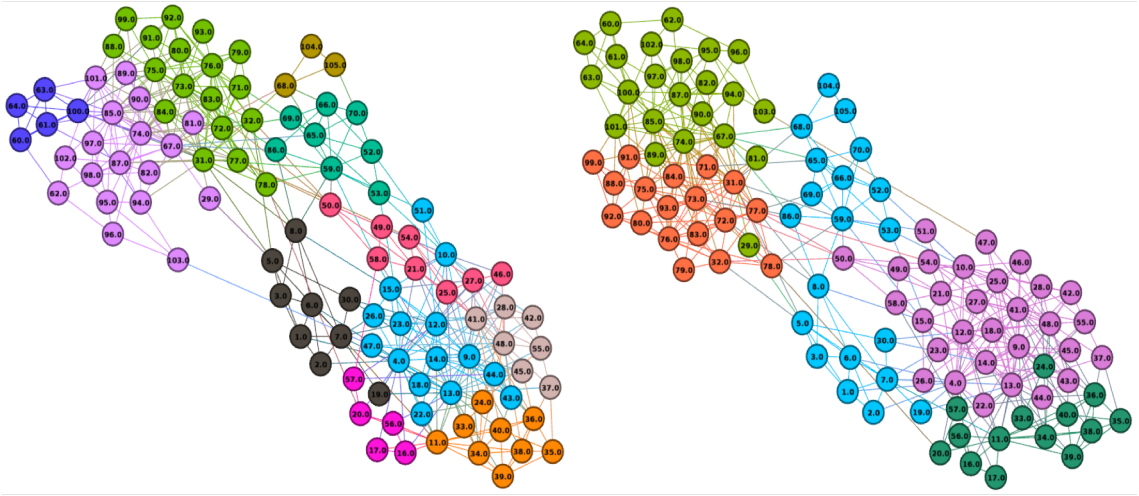


FIGURE 5.15. Two hierarchical levels detected in the Zachary Karate Club network, by tuning the parameter minimum community size K. Each color represents a community discovered by WMW.

(a) $K = 2$ (b) $K = 13$



(c) $K = 19$

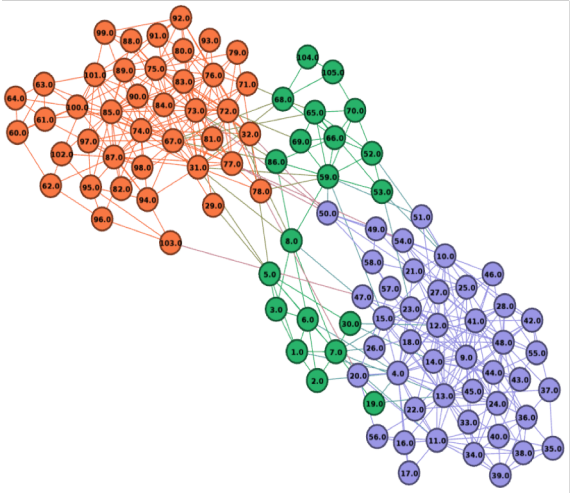


FIGURE 5.16. Three hierarchical levels detected in the Books about US Politics network, by tuning the parameter minimum community size K. Each color represents a community discovered by WMW.

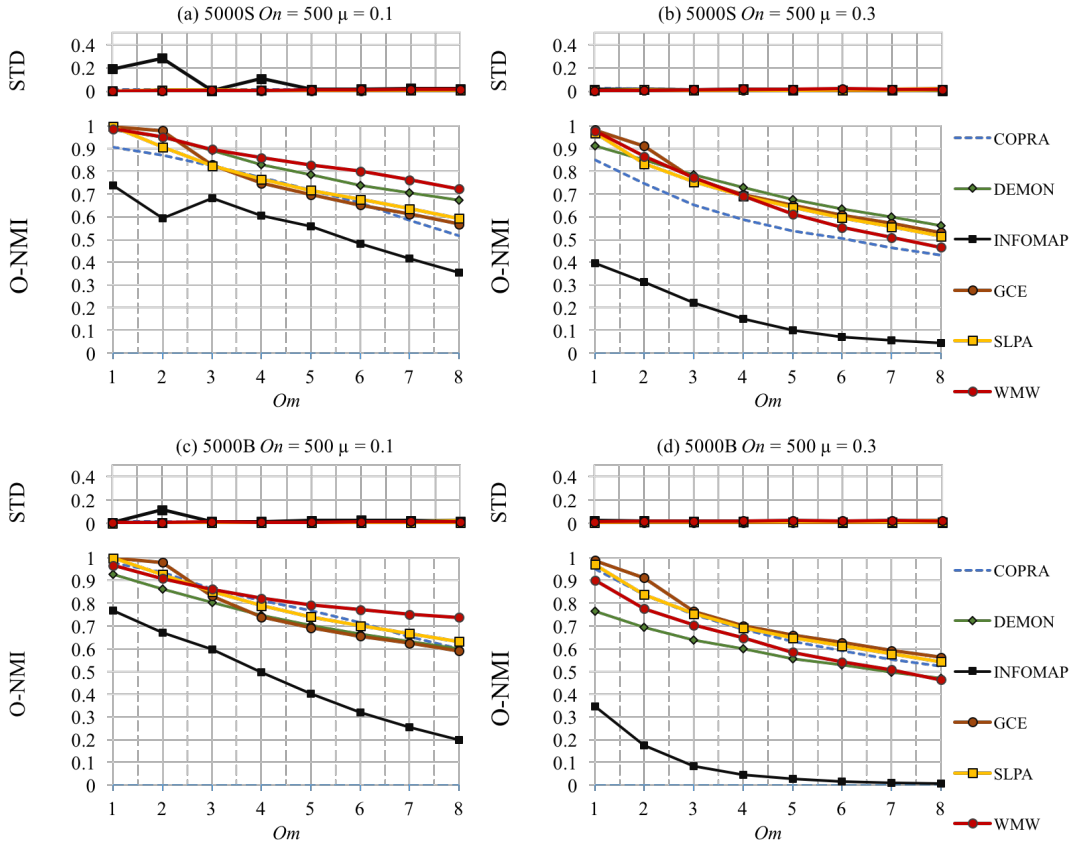


FIGURE 5.17. (Lower row) Mean value of the Overlapping Normalized Mutual Information, O-NMI (higher values are better) as function of the number of overlapping memberships, O_m. (Upper row) The standard deviation, STD (lower values are better) of the O-NMI as function of O_m.

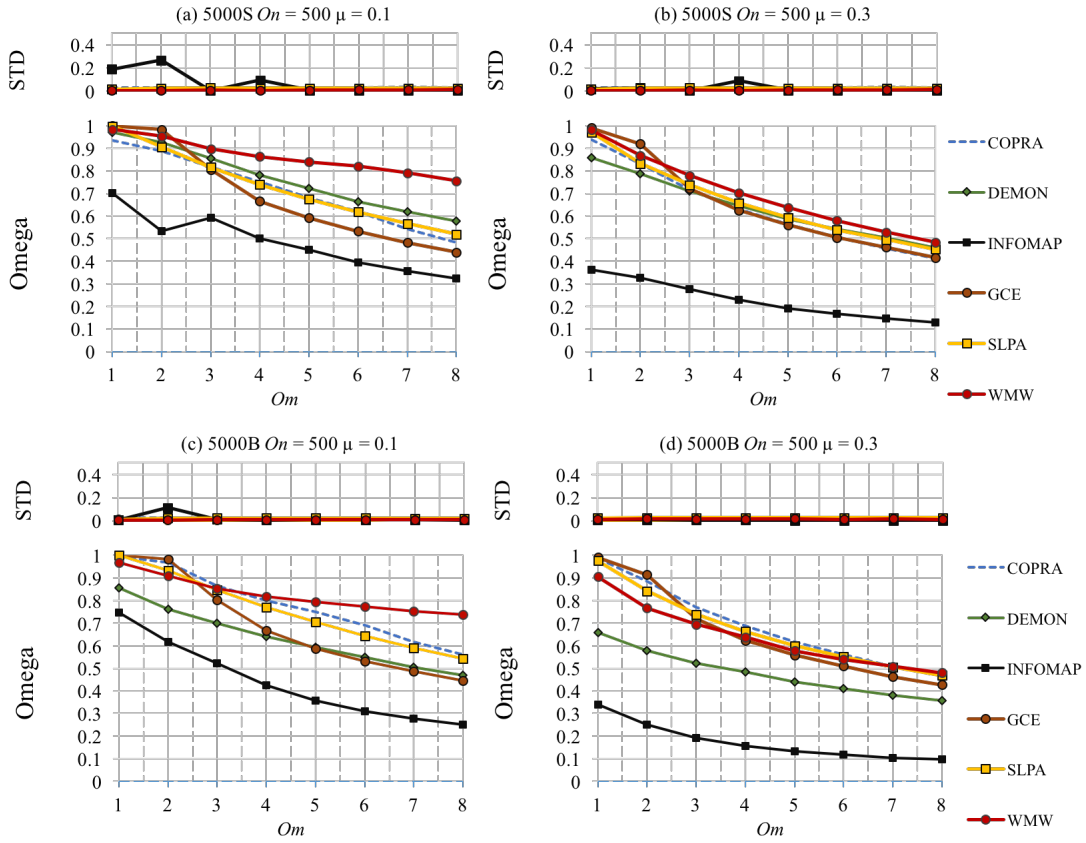


FIGURE 5.18. (Lower row) Mean value of the Adjusted Omega Index (higher values are better) as function of the number of overlapping memberships O_m. (Upper row) The standard deviation, STD (lower values are better) of the Adjusted Omega Index as function of O_m.

CHAPTER 6

Conclusions and Future Work

6.1 Conclusions

Several methods and algorithms have been proposed in the literature to perform community detection, however there are still important drawbacks and limitations that need to be solved. For example, the well known problem of resolution limit mostly present in algorithms that perform global optimization, also the unpredictable behavior presented by some algorithms due to the random nature of its functioning or the high dependency to the initial conditions, moreover the non-intuitive configuration of the algorithms that hinders its usability (lack of methodology to select the optimal parameters), and no less important the high computational complexity, that makes the algorithms impractical in large-scale scenarios, that are present nowadays on every real-world application (Big Data applications). In order to try to deal with the aforementioned drawbacks and limitations in community detection, in this thesis work have been proposed the following algorithms:

Dynamic Structural Similarity

- A novel Dynamic Structural Similarity on graphs was proposed. It determines the structural similarity of connected nodes in a graph by dynamically diffusing and capturing information beyond the immediate neighborhood of the analyzed nodes. This new similarity is modeled as an iterated function that can be solved by fixed point iteration in super-linear time and memory complexity, so it is able to analyze large-scale graphs.
- The dynamic structural similarity deals with the main drawback present in measures like Cosine, Jaccard, and Dice, that compute the similarities restricted to the immediate neighborhood of the nodes. Also, it deals with the high computational

complexity presented by the measures that go beyond the locality, such as the generalized edge clustering coefficient, among others.

- This new similarity exhibit interesting properties: *i)* By definition, it is self-contained and parameter free; *ii)* A dynamic interaction system emerges from it, because it exhibits many fluctuations for the similarities at early stages and converges to non-trivial steady-state as the system evolves. Moreover, it does not require a cooling factor in order to achieve convergence, as opposed to SimRank and PageRank based methods.

ISCAN Algorithm

- The advantages of the Dynamic Structural Similarity in the Community Detection task were evidenced with the proposed ISCAN algorithm. ISCAN is achieved after replacing the local structural similarity used in the SCAN algorithm with our proposed Dynamic Structural Similarity. Compared to the original SCAN algorithm, ISCAN outperforms the quality of the detected community structured, reduces its sensitivity to the parameter ϵ and maintains the same computational complexity, and those improvements are possible thanks to the back-end Dynamic Structural Similarity.

WMW Algorithm

- WMW is a novel fast heuristic algorithm for multi-scale and hierarchical community detection in complex networks inspired on an agglomerative hierarchical clustering technique, that merges communities with high Dynamic Structural Similarity. Through extensive experiments we show that our proposal can detect efficiently high quality community structure in complex networks, and compared to several state-of-the-art algorithms, WMW obtains superior results in several scenarios, so it can be considered a good candidate to perform community detection.
- WMW presents some interesting properties:
  - It achieves an average time complexity of $O(|E|)$ in complex networks. Such property makes it suitable to handle large-scale complex networks in a reasonable amount of time.
  - It can detect communities at any scale due to it does not possess a resolution limit, and it is able to detect hierarchical community structure.
  - It depends on two self-descriptive parameters, the minimum community size K and the community definition CD. These parameters with two default values if $(K = 2, CD = MOSTWEAK)$ are enough to detect a first hierarchy of communities, and further hierarchies can be found easily by tunning the parameter K with a clear proposed methodology.

  - It is unsupervised since it does not requires in advance the numbers of communities to be detected.
  - As opposed to the classical Agglomerative Hierarchical Clustering, our algorithm can achieve the convergence before a single cluster remains because it uses per-cluster quality functions instead of partition quality functions.

In a nutshell, the WMW algorithm offers better trade-off among quality of the results, computational complexity and usability, compared to the state-of-the-art.

Overlapping WMW Algorithm (OWMW)

- The OWMW algorithm is an extension of the WMW algorithm to detect fuzzy and crisp community structure. The OWMW algorithm leverages the disjoint community structure generated by WMW to build efficiently an overlapping community structure. Three core elements have been proposed to compute the overlapping community structure: *i)* The connectivity function that quantifies the density of connections of a node towards a particular disjoint community, relying its computation on the Dynamic Structural Similarity measure. *ii)* The ϵ-Core community definition that increases the probability of identifying in-between communities in the disjoint community structure. *iii)* The membership function to compute the soft partition from the core disjoint communities.
- The experimental evaluation shows that our proposal can detect efficiently high quality overlapping community structure in complex networks, and compared to several state-of-the-art algorithms, OWMW obtains superior results in several scenarios, so it can be considered a good candidate to perform overlapping community detection. The OWMW algorithm keeps the same computational complexity of the original WMW algorithm, thus it is still applicable to large-scale graphs. Additionally, OWMW requires as extra parameter a crisp threshold (real value in the interval [0, 1]) in case crisp overlapping structure is desired.

6.2 Future Work

The following directions of future work are proposed to further develop this thesis:

- This research work was focused on community detection in unweighted undirected graphs. Further research should be addressed to adapt the algorithms, if possible, to other types of graphs, e.g., weighted, directed and bipartite graphs. In fact, the proposed algorithms can be adapted easily to weighted graphs, but they need to be tested. However, adapting the algorithms to directed graphs is a challenging task, due to the asymmetric relationship between the nodes.

- Further research should focus on adapting the proposed algorithms to dynamic graphs generated in data stream models. This is a promising research work since the proposed algorithms works in a local basis, and that is a *must be* characteristic of algorithms to process data streams.

- It could be possible to implement the proposed algorithms on parallel and/or distributed architectures. We guess that the dynamic structural similarity and the WMW algorithm are susceptible to achieve up-to linear speed-ups in such architectures since their core computations can be carried independently in multiple parallel/distributed processing units.

- We concluded that the WMW algorithm is a good candidate to perform disjoint/overlapping community detection. For that reason, it could be interesting to see this algorithm hands-on in a real case of study (e.g., analysis of social networks) or in engineering applications (e.g., recommendation systems, tracking systems, anomaly detection).

Bibliography

[1] Emmanuel Abbe, *Community detection and stochastic block models: recent developments*, CoRR **abs/1703.10146** (2017).

[2] Lada A. Adamic, Rajan M. Lukose, Amit R. Puniyani, and Bernardo A. Huberman, *Search in power-law networks*, Physical review. E, Statistical, nonlinear, and soft matter physics **64 4 Pt 2** (2001), 046135.

[3] William Aiello, Fan Chung, and Linyuan Lu, *A random graph model for massive graphs*, Proceedings of the Thirty-second Annual ACM Symposium on Theory of Computing, ACM, 2000, pp. 171–180.

[4] David A Bader, Shiva Kintali, Kamesh Madduri, and Milena Mihail, *Approximating betweenness centrality*, International Workshop on Algorithms and Models for the Web-Graph, Springer, 2007, pp. 124–137.

[5] Albert-László Barabási and Réka Albert, *Emergence of scaling in random networks*, science **286** (1999), no. 5439, 509–512.

[6] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre, *Fast unfolding of communities in large networks*, Journal of statistical mechanics: theory and experiment **2008** (2008), no. 10, P10008.

[7] Ulrik Brandes, *A faster algorithm for betweenness centrality*, Journal of mathematical sociology **25** (2001), no. 2, 163–177.

[8] ———, *On variants of shortest-path betweenness centrality and their generic computation*, Social Networks **30** (2008), no. 2, 136–145.

[9] S. Brin and L. Page, *The anatomy of a large-scale hypertextual web search engine*, Seventh International World-Wide Web Conference (WWW 1998), 1998.

[10] Eduar Castrillo, Elizabeth León, and Jonatan Gómez, *Fast heuristic algorithm for multi-scale hierarchical community detection*, Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017, ACM, 2017, pp. 982–989.

[11] Tanmoy Chakraborty, *Leveraging disjoint communities for detecting overlapping community structure*, CoRR **abs/1504.06608** (2015).

[12] Lijun Chang, Wei Li, Lu Qin, Wenjie Zhang, and Shiyu Yang, *pscan: Fast and exact structural graph clustering*, IEEE Transactions on Knowledge and Data Engineering **29** (2017), no. 2, 387–401.

[13] Jie Chen and Yousef Saad, *Dense subgraph extraction with application to community detection*, IEEE Transactions on Knowledge and Data Engineering **24** (2012), no. 7, 1216–1230.

[14] Norishige Chiba and Takao Nishizeki, *Arboricity and subgraph listing algorithms*, SIAM Journal on Computing **14** (1985), no. 1, 210–223.

[15] Aaron Clauset, Mark EJ Newman, and Cristopher Moore, *Finding community structure in very large networks*, Physical review E **70** (2004), no. 6, 066111.

[16] A F Cohen, Erez, ben Avraham, and Havlin, *Resilience of the internet to random breakdowns*, Physical review letters **85 21** (2000), 4626–8.

[17] Yehonatan Cohen, Danny Hendler, and Amir Rubin, *Node-centric detection of overlapping communities in social networks*, International Conference and School on Network Science, Springer, 2017, pp. 1–10.

[18] L. M. Collins and Colin Dent, *Omega: A general formulation of the rand index of cluster recovery suitable for non-disjoint solutions.*, Multivariate behavioral research **23 2** (1988), 231–42.

[19] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein, *Introduction to algorithms, third edition*, 3rd ed., The MIT Press, 2009.

[20] Michele Coscia, Giulio Rossetti, Fosca Giannotti, and Dino Pedreschi, *Demon: a local-first discovery method for overlapping communities*, Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2012, pp. 615–623.

[21] Gabor Csardi and Tamas Nepusz, *The igraph software package for complex network research*, InterJournal, Complex Systems **1695** (2006), no. 5, 1–9.

[22] Stijn Dongen, *A cluster algorithm for graphs*, (2000).

[23] Kayhan Erciyes, *Complex networks: an algorithmic perspective*, CRC Press, 2014.

[24] P. Erdos and A. Rényi, *On the existence of a factor of degree one of a connected random graph*, Acta Mathematica Hungarica **17** (1966), no. 3-4, 359–368.

[25] P. Erdős and A Rényi, *On the evolution of random graphs*, PUBLICATION OF THE MATHEMATICAL INSTITUTE OF THE HUNGARIAN ACADEMY OF SCIENCES, 1960, pp. 17–61.

[26] Santo Fortunato and Marc Barthelemy, *Resolution limit in community detection*, Proceedings of the National Academy of Sciences **104** (2007), no. 1, 36–41.

[27] Santo Fortunato and Darko Hric, *Community detection in networks: A user guide*, Physics Reports **659** (2016), 1–44.

[28] Michelle Girvan and Mark EJ Newman, *Community structure in social and biological networks*, Proceedings of the national academy of sciences **99** (2002), no. 12, 7821–7826.

[29] Jonatan Gomez, Dipankar Dasgupta, and Olfa Nasraoui, *A new gravitational clustering algorithm*, Proceedings of the 2003 SIAM International Conference on Data Mining, SIAM, 2003, pp. 83–94.

[30] M.S. Granovetter, *The Strength of Weak Ties*, The American Journal of Sociology **78** (1973), no. 6, 1360–1380.

[31] Steve Gregory, *Finding overlapping communities in networks by label propagation*, New Journal of Physics **12** (2010), no. 10, 103018.

[32] Jihui Han, Wei Li, and Weibing Deng, *Multi-resolution community detection in massive networks*, Scientific Reports **6** (2016), 38998 EP –.

[33] Alexandre Hollocou, Thomas Bonald, and Marc Lelarge, *Improving pagerank for local community detection*, arXiv preprint arXiv:1610.08722 (2016).

[34] Yanqing Hu, Hongbin Chen, Peng Zhang, Menghui Li, Zengru Di, and Ying Fan, *Comparative definition of community and corresponding identifying algorithm*, Physical Review E **78** (2008), no. 2, 026121.

[35] Jianbin Huang, Heli Sun, Jiawei Han, Hongbo Deng, Yizhou Sun, and Yaguang Liu, *Shrink: a structural clustering algorithm for detecting hierarchical communities in networks*, Proceedings of the 19th ACM international conference on Information and knowledge management, ACM, 2010, pp. 219–228.

[36] Glen Jeh and Jennifer Widom, *Simrank: a measure of structural-context similarity*, Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2002, pp. 538–543.

[37] Pablo Jensen, Matteo Morini, Márton Karsai, Tommaso Venturini, Alessandro Vespignani, Mathieu Jacomy, Jean-Philippe Cointet, Pierre Mercklé, and Eric Fleury, *Detecting global bridges in networks*, Journal of Complex Networks **4** (2015), no. 3, 319–329.

[38] Weibing Den Jihui Han, Wei Li, *Multi-scale community detection in massive networks*, IEEE Transactions on Knowledge and Data Engineering **24** (2012), no. 7, 1216–1230.

[39] Ruoming Jin, Victor E. Lee, and Hui Hong, *Axiomatic ranking of network role similarity*, KDD, 2011.

[40] Tatsuro Kawamoto and Martin Rosvall, *Estimating the resolution limit of the map equation in community detection.*, Physical review. E, Statistical, nonlinear, and soft matter physics **91 1** (2015), 012809.

[41] Kyle Kloster and David F Gleich, *Heat kernel based community detection*, Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2014, pp. 1386–1395.

[42] V. Krebs, *A network of co-purchased books about us politics sold by the online bookseller amazon.com*, (2008).

[43] Michael Krivelevich, *Triangle factors in random graphs*, Comb. Probab. Comput. **6** (1997), no. 3, 337–347.

[44] Deepika Lalwani, Durvasula VLN Somayajulu, and P Radha Krishna, *A community driven social recommendation system*, Big Data (Big Data), 2015 IEEE International Conference on, IEEE, 2015, pp. 821–826.

[45] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi, *Benchmark graphs for testing community detection algorithms*, Physical review E **78** (2008), no. 4, 046110.

[46] Conrad Lee, Fergal Reid, Aaron McDaid, and Neil Hurley, *Detecting highly overlapping community structure by greedy clique expansion*, arXiv preprint arXiv:1002.1827 (2010).

[47] Jure Leskovec and Andrej Krevl, *SNAP Datasets: Stanford large network dataset collection*, `http://snap.stanford.edu/data`, June 2014.

[48] Yixuan Li, Kun He, David Bindel, and John E Hopcroft, *Uncovering the small community structure in large networks: a local spectral approach*, Proceedings of the 24th international conference on world wide web, International World Wide Web Conferences Steering Committee, 2015, pp. 658–668.

[49] Peter Lofgren, Siddhartha Banerjee, and Ashish Goel, *Personalized pagerank estimation and search: A bidirectional approach*, Proceedings of the Ninth ACM International Conference on Web Search and Data Mining, ACM, 2016, pp. 163–172.

[50] Aaron F McDaid, Derek Greene, and Neil Hurley, *Normalized mutual information to evaluate overlapping community finding algorithms*, arXiv preprint arXiv:1110.2515 (2011).

[51] Mark Newman, Albert-Laszlo Barabasi, and Duncan J Watts, *The structure and dynamics of networks*, Princeton University Press, 2011.

[52] Mark EJ Newman, *The structure and function of complex networks*, SIAM review **45** (2003), no. 2, 167–256.

[53] Mark EJ Newman and Michelle Girvan, *Finding and evaluating community structure in networks*, Physical review E **69** (2004), no. 2, 026113.

[54] Ferran Parés, Dario Garcia Gasulla, Armand Vilalta, Jonatan Moreno, Eduard Ayguadé, Jesús Labarta, Ulises Cortés, and Toyotaro Suzumura, *Fluid communities: A competitive, scalable and diverse community detection algorithm*, International Workshop on Complex Networks and their Applications, Springer, 2017, pp. 229–240.

[55] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, *Scikit-learn: Machine learning in Python*, Journal of Machine Learning Research **12** (2011), 2825–2830.

[56] Pascal Pons and Matthieu Latapy, *Computing communities in large networks using random walks*, International symposium on computer and information sciences, Springer, 2005, pp. 284–293.

[57] Filippo Radicchi, Claudio Castellano, Federico Cecconi, Vittorio Loreto, and Domenico Parisi, *Defining and identifying communities in networks*, Proceedings of the National Academy of Sciences of the United States of America **101** (2004), no. 9, 2658–2663.

[58] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara, *Near linear time algorithm to detect community structures in large-scale networks*, Physical review E **76** (2007), no. 3, 036106.

[59] Anatol Rapoport, *Contribution to the theory of random and biased nets*, Bulletin of Mathematical Biology **19** (1957), 257–277.

[60] Erzsébet Ravasz and Albert-László Barabási, *Hierarchical organization in complex networks*, Physical Review E **67** (2003), no. 2, 026112.

[61] Matteo Riondato and Eli Upfal, *Abra: Approximating betweenness centrality in static and dynamic graphs with rademacher averages*, Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 1145–1154.

[62] Ryan Rossi and Nesreen Ahmed, *The network data repository with interactive graph analytics and visualization.*, AAAI, vol. 15, 2015, pp. 4292–4293.

[63] Martin Rosvall and Carl T Bergstrom, *Maps of random walks on complex networks reveal community structure*, Proceedings of the National Academy of Sciences **105** (2008), no. 4, 1118–1123.

[64] Junming Shao, Zhichao Han, and Qinli Yang, *Community detection via local dynamic interaction*, arXiv preprint arXiv:1409.7978 (2014).

[65] Hiroaki Shiokawa, Yasuhiro Fujiwara, and Makoto Onizuka, *Scan++: efficient algorithm for finding clusters, hubs and outliers on large-scale graphs*, Proceedings of the VLDB Endowment **8** (2015), no. 11, 1178–1189.

[66] Victor Spirin and Leonid A Mirny, *Protein complexes and functional modules in molecular networks*, Proceedings of the National Academy of Sciences **100** (2003), no. 21, 12123–12128.

[67] Jing Wang and Ioannis Ch Paschalidis, *Botnet detection based on anomaly and community detection*, IEEE Transactions on Control of Network Systems **4** (2017), no. 2, 392–404.

[68] Duncan J Watts and Steven H Strogatz, *Collective dynamics of 'small-world'networks*, nature **393** (1998), no. 6684, 440.

[69] Dong Wen, Lu Qin, Ying Zhang, Lijun Chang, and Xuemin Lin, *Efficient structural graph clustering: An index-based approach*, Proceedings of the VLDB Endowment **11** (2017), no. 3, 243–255.

[70] Jierui Xie, Stephen Kelley, and Boleslaw K Szymanski, *Overlapping community detection in networks: The state-of-the-art and comparative study*, Acm computing surveys (csur) **45** (2013), no. 4, 43.

[71] Jierui Xie, Boleslaw K Szymanski, and Xiaoming Liu, *Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process*, Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on, IEEE, 2011, pp. 344–349.

[72] Xiaowei Xu, Nurcan Yuruk, Zhidan Feng, and Thomas AJ Schweiger, *Scan: a structural clustering algorithm for networks*, Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2007, pp. 824–833.

[73] Zhao Yang, René Algesheimer, and Claudio J Tessone, *A comparative analysis of community detection algorithms on artificial networks*, Scientific Reports **6** (2016), 30750.

[74] Nurcan Yuruk, Mutlu Mete, Xiaowei Xu, and Thomas AJ Schweiger, *Ahscan: Agglomerative hierarchical structural clustering algorithm for networks*, Social Network Analysis and Mining, 2009. ASONAM'09. International Conference on Advances in, IEEE, 2009, pp. 72–77.

[75] Wayne W Zachary, *An information flow model for conflict and fission in small groups*, Journal of anthropological research **33** (1977), no. 4, 452–473.